

روش کاربردی

تحلیل

نیازمندی‌های نرم افزار

نویسنده‌ان:

یوسف مهرداد بی‌بالان
پویا شبازیان
مصطفی ایراف

ویراستار:

ابراهیم نقیب‌زاده مشایخ
عضو هیأت علمی دانشکده علوم کامپیوتر دانشگاه تهران



روش کاربردی تحلیل نیازمندی‌های نرم‌افزار

نویسنده‌گان:

یوسف مهرداد بی‌بالان

پویا شهبازیان

مصطفوی ایراف

ویراستار:

ابراهیم نقیب‌زاده مشایخ

عضو هیأت علمی دانشکده علوم کامپیوتر دانشگاه تهران

عنوان و نام پدیدآور	: مهرداد بی بالان، یوسف، ۱۳۵۳ - روش کاربردی تحلیل نیازمندی‌های نرمافزار / نویسندهان یوسف مهرداد بی بالان، پویا شهبازیان، مظفر ایراف؛ ویراستار ابراهیم نقیبزاده مشایخ.
مشخصات نشر	: تهران: صفار، ۱۳۹۲
مشخصات ظاهری	: ۲۷۲ ص. مصور، جدول، نمودار.
شابک	: ۹۷۸-۹۶۴-۳۸۸-۴۰۰-۰
وضعیت فهرست نویسی:	فیبا
یادداشت	: چاپ قبلی: رسم، ۱۳۸۹ (۳۵۹ ص)
یادداشت	: واژنامه.
یادداشت	: کتابنامه.
موضوع	: نرمافزار -- تولید
شناسه افزوده	: شهبازیان، پویا، ۱۳۶۰ -
شناسه افزوده	: ایراف، مظفر، ۱۳۵۳ -
شناسه افزوده	: نقیب زاده مشایخ، ابراهیم، ویراستار
رده‌بندی کنگره	: QA۷۶/۷۶ ت۹۹۱۳۹۲
رده‌بندی دیوبی	: ۰۰۵/۱۱
شماره کتابشناسی ملی	: ۳۲۷۴۳۶۰

فهرستنویسی پیش از انتشار: انتشارات صفار



۳۰۰۰۵۳۵۱

نام کتاب	: روش کاربردی تحلیل نیازمندی‌های نرمافزار
تألیف	: یوسف مهرداد بی بالان، پویا شهبازیان، مظفر ایراف
ویراستار	: ابراهیم نقیبزاده مشایخ
معرفت	: گنج شایگان ۵۵۴۰۲۱۸۴ ①
لیتوگرافی	: گنج شایگان ۵۵۴۰۳۴۷۸ ①
چاپخانه	: ناظر چاپ فنی: اصغر امیرشکری
شمارگان	: شماره ۱۱۰۰
نوبت چاپ	: دوم- پاییز ۱۳۹۶
قیمت	: ۱۵۰۰۰ ریال
ناشر	: انتشارات صفار
مرکز پخش	: خیابان انقلاب- روبروی دبیرخانه دانشگاه تهران- بازارچه کتاب- طبقه همکف
	: انتشارات اشراقی ۶۶۴۰۸۴۸۷ ②
	: تلفن: ۶۶۹۷۰۹۹۲
	: خیابان انقلاب- روبروی دبیرخانه دانشگاه تهران- بازارچه کتاب- طبقه زیرین
	: پخش کتاب بینش ۶۶۴۹۶۲۹۹ ③
	: کتابفروشی صفا ۶۶۹۷۸۸۴۶ ④

www.saffarpublishing.ir

www.Eshraghi.ir

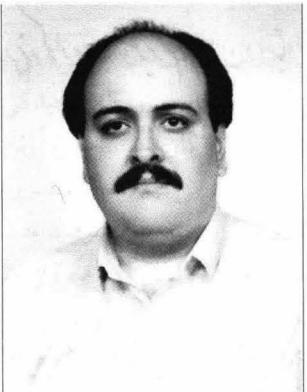
Email: saffar_publishing@yahoo.com

شابک : ۹۷۸-۹۶۴-۳۸۸-۴۰۰-۰

ISBN 978-964-388-400-0

این اثر، مشمول قانون حمایت مؤلفان و مصنفان و هنرمندان مصوب ۱۳۴۸ است، هر کس تمام یا قسمتی از این اثر را بدون اجازه مؤلف (ناشر) نشر، یا پخش یا عرضه کند مورد پیگرد قانونی قرار خواهد گرفت.

مدیریت واحد تولید انتشارات صفار: ۰۹۱۲-۱۰۷۳۰۰۳



تندیم به:
دکتر رامن راسین
استاد کران قدر و داشمند فریخته

تّعديم به سرمه باشم که زمان صرف شده برای اين کتاب متعلق به او بود.

پویا شهبازیان

تّعديم به روح پدر بزرگوارم، مادر عزيرم، معلمان و استادان کراقدرم، همسر صبورم و فرزندان دلندم، فاطمه و امير حسین
طفیر ايراف

به ياد پدر، مادر و تّعديم به دوست، همراه و همسر همراهان و بالکه ذشم، روا

يوسف مردادي بالان

مقدمه ۴

حوزه نیازمندی‌های نرم‌افزاری یکی از مهم‌ترین حوزه‌ها در توسعه نرم‌افزار است و با وجود پیشرفت‌های چشم‌گیر، مهندسی نیازمندی‌ها جزو ضعیف‌ترین حلقه‌ها در زنجیره فعالیت‌های مهندسی نرم‌افزار است.

عموماً فرایند توسعه نرم‌افزار در شرکت‌های ایرانی - کوچک یا بزرگ - با چالش‌های زیادی همراه است. گاه توسعه فارغ از اصول علمی و مهندسی انجام می‌گردد و گاه در ورطه پیچیدگی متداول‌وزی‌ها فرو می‌رود. در این شرایط، متداول‌وزی به جای راهگشاپی، خود گره کار است. عموماً این دو چالش به دلیل نبود برداشتی ساده اما عمیق از اصول و مفاهیم مهندسی ایجاد می‌گردد. اغلب رعایت همین اصول و مفاهیم به دور از رویرو شدن با پیچیدگی‌های متداول‌وزی یا امید به پیشرفت تصادفی پرورده، آن را به سرانجامی موفق خواهد رساند. این کتاب سعی دارد برخی از این مفاهیم و اصول را تبیین کند.

یکی دیگر از عوامل بروز چالش‌های مذکور، دشواری نگاشت و اجرای فعالیت‌های فرآیندهای RUP در تیم‌ها است. این عامل باعث می‌شود که چارچوب فکری تیم‌ها به جای وظیفه‌محوری (Task Driven) به محصول‌محوری (Work Product Driven) سوق داده شود. به عنوان مثال، تیم توسعه تنها به تدوین محصول‌کاری «سنند چشم‌انداز» می‌اندیشد، نه به انجام مجموعه‌ای از وظیفه‌ها و کارها که منجر به تهیه این سنند می‌شود. چگونگی انجام وظیفه‌ها، ترتیب اجرای آن‌ها و تکنیک‌های مرتبط، منجر به شکل‌گیری چارچوبی ذهنی در تیم می‌گردد که در روش محصول‌محوری کم‌رنگ است. این کتاب در کنار معرفی محصولات کاری، بر وظیفه‌ها و تکنیک‌های انجام آن‌ها نیز تأکید دارد.

هر چند مبنای اصلی روش ارائه شده در این کتاب RUP است، اما دارای تفاوت‌هایی از قبیل جایه‌جایی، تغییر جزئیات، ایجاد توالی و حذف ارتباط گرافی بین فعالیت‌ها، تغییر تعداد و محتوای محصولات کاری و بازطرابی روش انجام فعالیت‌ها است. لذا این کتاب به سه بخش «تحلیل مسئله»، «شناسایی نیازهای ذینفعان» و «تعریف سیستم» در نیازمندی‌ها پای‌بند بوده است. تغییرات یادشده ماحصل اجرای چندین باره این فرایند در پروژه‌ها و دوره‌های تخصصی است. باگذشت زمان، مسائلی که در تیم‌های توسعه نرم‌افزار با آن‌ها مواجه بودیم در حال تغییر بودند و

تمرکز بر حل آن‌ها باعث فراموشی تجارب گذشته می‌گردید. فراموشی تجارب گذشته که می‌توانست برای تیم‌های دیگر راه‌گشا و کاربردی باشد - نگران‌کننده بود. این کتاب تلاشی برای تدوین آموخته‌ها و انتقال آن‌ها به خوانندۀ علاقه‌مند است.

با گذشت زمان و رشد پیچیدگی سیستم‌های نرم‌افزاری و به دنبال آن رشد پیچیدگی اجرای پروژه‌ها، لازم است اعضای تیم -خصوصاً رهبران- برداشت یکسانی از موضوعات نیازمندی‌ها داشته باشند تا مذاکره و تعامل برای حل مسأله‌های پیچیده امکان‌پذیر باشد. اغلب، داشش و آشنایی اعضای تیم‌ها با مفاهیم، تکنیک‌ها و روش‌های مهندسی نیازمندی‌ها کفایت نمی‌کند. آموزش، راه طولانی و زمان‌بری است که اغلب در شرایط اضطراری پروژه‌ها، راه حل مناسبی نیست. این کتاب راه میان‌بری برای تحقق اهداف یادشده است.

حوزه نیازمندی‌های نرم‌افزاری یکی از وسیع‌ترین حوزه‌ها در مهندسی نرم‌افزار است که استخراج، تحلیل، توصیف، تصدیق، فرایند و ماهیت تدریجی این، مدیریت تغییرات، صفات، ردبایی و اندازه‌گیری نیازمندی‌ها را شامل می‌شود که در پیکره دانش مهندسی نرم‌افزار (Software Engineering) (Body of Knowledge) بدان‌ها اشاره شده است. از طرف دیگر پیکره‌دانش تحلیل کسب‌وکار (Business Analysis Body of Knowledge) دارای چارچوب مشخصی است که تحلیل سازمان، استخراج و تحلیل نیازمندی‌ها، تصدیق و ارزشیابی راه حل را دربر می‌گیرد. با هدف ارائه کتابی کاربردی، ساده و کم حجم، تمام حوزه‌های مرتبط با مهندسی نیازمندی‌ها، مدل‌سازی کسب‌وکار و نکات فرایندی مربوط به آن‌ها در کتاب آورده نشده است. امید است در کتاب‌های بعدی این مهم تحقیق یابد.

کتاب برای استفاده در پروژه‌های توسعه نرم‌افزار تدوین شده است، لذا دیدگاه کاربردی بر دیدگاه ادبیات حوزه‌ای در آن غالب است، اما مفاهیم و اصطلاحات مورد نیاز برای کاربرد در آن گنجانده شده است.

معادل واژه‌های لاتین از لغت‌نامه تخصصی رایانه و فناوری اطلاعات فرهنگستان زبان و ادب فارسی برگرفته شده است و برای واژه‌هایی که در این لغت‌نامه وجود نداشتند، معادل‌های فارسی برگزیده شده است.

برای ارتباط با ما می‌توانید از نشانی الکترونیکی info@somamos-co.ir یا وب‌گاه www.somamos-co.ir استفاده کنید.

مخاطبان کتاب

مخاطبان اصلی کتاب، تحلیل‌گران سیستم‌های نرم‌افزاری هستند. به علاوه کتاب می‌تواند برای گروه‌های زیر نیز مفید باشد:

مشتریان و کارفرمایان پروژه‌های توسعه نرم‌افزار
یکی از مهم‌ترین مشخصه‌های مرحله نیازمندی‌ها، تدوین مشخصات سیستم به زبان مشتریان است. این کتاب به مشتریان می‌آموزد که چه چیزهایی را و چگونه از تیم‌های توسعه نرم‌افزار درخواست کنند و به چه نکاتی در این مرحله توجه کنند.

طراحان و برنامه‌نویسان نرم‌افزار

خرنگی‌های مرحله نیازمندی‌ها، ورودی به تیم‌های طراحی و برنامه‌نویسی است. همچنین طراحان نرم‌افزار در بازنگری و تأیید خروجی‌های این مرحله مشارکت مؤثر دارند. این کتاب به آن‌ها کمک خواهد کرد تا وظایف خود را در مرحله نیازمندی‌ها به درستی انجام دهند.

مدیران پروژه‌های توسعه نرم‌افزار

این کتاب چارچوبی برای برنامه‌ریزی مرحله نیازمندی‌ها می‌بینی بر فعالیت‌ها و محصولات کاری با قالب‌های مشخص ارائه می‌دهد. این چارچوب برای برنامه‌ریزی بخشی از پروژه بسیار مفید خواهد بود. تبیین جایگاه محصولات کاری در مرحله نیازمندی‌ها، ریسک‌های مشخصی را به مدیران پروژه یادآوری می‌کند.

آزمون‌گران نرم‌افزار

شناخت ورودی‌های هر فعالیت، یکی از عوامل مهم در انجام درست آن است. مجموعه نیازمندی‌های سیستم نیز ورودی اصلی آزمون سیستم‌های نرم‌افزاری است. این کتاب خروجی‌های مرحله نیازمندی‌ها –ورودی‌های مرحله آزمون– و اجزای هر یک از آن‌ها را به آزمون‌گران می‌آموزد.

دانشجویان رشته‌های مهندسی نرم‌افزار و رشته‌های مرتبط

مهندسي نیازمندی‌ها بخش مهمی از مهندسی نرم‌افزار است. با وجود مطالعه هم‌چون جایگاه نیازمندی‌ها در توسعه نرم‌افزار، تعاریف و مفاهیم اصلی، چالش‌ها و تکنیک‌های استخراج و هم‌چنین گام‌های تحلیل نیازمندی‌ها در یک ساختار نظاممند، این کتاب منبع مناسبی برای دانشجویان خواهد بود.

بخش‌ها و فصل‌های کتاب

کتاب در شش بخش ارائه شده است و بخش‌ها بر پایه ارتباط و پیوستگی مطالب چیده شده‌اند. بخش ابتدایی به اهمیت و تعاریف اختصاص دارد، روش اجرای فرایند در بخش‌های میانی قرار گرفته است و دو بخش پایانی نیز شامل مطالب تکمیلی است. در ادامه، بخش‌ها و فصل‌های کتاب معرفی می‌شود.

بخش اول: مقدمه

این بخش مقدمه‌ای بر موضوعات حوزه نیازمندی‌هاست که در دو فصل ارائه می‌شود.

فصل اول: اهمیت نیازمندی‌ها

بی‌شک مرحله نیازمندی‌ها یکی از مهم‌ترین و تأثیرگذارترین مراحل در توسعه نرم‌افزار است. این فصل با ارائه و تأکید بر نتایج تحقیق‌های انجام شده، اهمیت و تأثیرگذاری نیازمندی‌ها را در موقیت پروژه‌ها بررسی می‌کند.

منبع اصلی این فصل:

Dean Leffingwell and Don Widrig, "Managing software requirements: A Use Case Approach", Addison-Wesley, 2003

فصل دوم: تعاریف

واژه‌ها و مفاهیم نقش مهمی در درست آموزه‌ها و بهبود ارتباطات انسانی دارند. به کارگیری واژه‌هایی چون «نیاز» یا «نیازمندی» به جای یکدیگر در گفت‌وگوها و مکتوبات امر رایجی است، اما در گفتارها و نوشتارهای تخصصی، واژه‌ها معنای ویژه‌ای می‌گیرند تا آن‌جا که دیگر نمی‌توان «نیاز» را به جای «نیازمندی» به کار برد.

در این فصل، مفاهیم، واژه‌ها و اصطلاحات مهم حوزه نیازمندی‌ها تعریف شده است.

بخش دوم: تحلیل مسأله

استخراج و تحلیل نیازمندی‌ها در پروژه‌های توسعه نرم‌افزار شباهت زیادی به حل مسائل ریاضی دارد. در کتاب‌های آموزش ریاضی تأکید می‌شود که فهم درست مسأله، نیمی از حل آن است. این بخش از کتاب به تشریح چگونگی فهم مسأله در پروژه‌ها می‌پردازد که شامل یک فصل است.

فصل سوم: روش تحلیل مسأله

در این فصل بعد از تعریف واژه «مسأله»، اهمیت شناسایی درست مسأله توضیح داده شده است. سپس روش شناسایی و تدوین مسأله، افراد مرتبط با مسأله یا ذینفعان، مرز سیستم و قیدهای محدودکننده راه حل تشریح شده‌اند.

بخش سوم: شناسایی نیازهای ذینفعان

هدف این بخش شناسایی نیازهای ذینفعان پروژه و تدوین مشخصات محصول مورد انتظار و مطلوب آنان است. علاوه بر شناسایی نیازهای ذینفعان، قواعد حاکم بر کسب‌وکار نیز استخراج و مدون می‌گردد تا در تعریف سیستم استفاده شوند. این بخش شامل دو فصل است.

فصل چهارم: روش شناسایی نیازهای ذینفعان

این فصل مجموعه‌ای از کارها را شرح می‌دهد که منجر به تدوین سند چشم‌انداز سیستم می‌گردد. این کارها به طور کلی شامل تشریح جایگاه محصول نزد ذینفعان، شناسایی نیازها و شرح چگونگی رفع آن‌ها توسط سیستم، شناسایی ویژگی‌ها، اولویت آن‌ها و تعیین مستندات الزامی برای سیستم است.

فصل پنجم: شناسایی قواعد کسب و کار

قواعد کسب و کار نقش بارزی در توسعه سیستم‌های نرم‌افزاری ایفا می‌کنند، چرا که سیستم ملزم به پوشش بخش عملهای از آن‌هاست. در این فصل ابتدا تعریفی از قواعد کسب و کار، خاستگاه، طبقه‌بندی و سطح‌بندی آن‌ها ارائه شده است. در ادامه، روش شناسایی و مستندسازی قواعد در قالب نمودارهای UML و مستندات تشریح شده است.

بخش چهارم: تعریف سیستم

تعیین مشخصات سیستم در سطح کلان که ویژگی‌ها (سرویس‌ها) بخشی از آن است، پیش‌تر مطرح شد. ویژگی‌ها فقط بیان می‌کنند که «چه» سرویس‌هایی توسط سیستم ارائه می‌شود، اما «چگونگی» ارائه آن‌ها را بیان نمی‌کنند.

این بخش نشان می‌دهد که سیستم «چگونه» نیازها را مرتفع و ویژگی‌ها را ارائه می‌کند. در این کتاب، برای این کار از تکنیک موردکاربرد (use case) استفاده شده است. این بخش شامل سه فصل است.

فصل ششم: مدل موردکاربرد

در این فصل اجزای مدل موردکاربرد و محصولات کاری مرتبط با آن معرفی شده‌اند. تعریف، اهمیت و روش‌های طبقه‌بندی کنشگر (actor) و موردکاربرد، اجزای مستندات «مشخصات موردکاربرد» (use case specification) و «مروار مدل موردکاربرد» (use case model survey) از جمله مواردی است که در این فصل تشریح می‌شود.

فصل هفتم: تدوین مدل موردکاربرد

تدوین مدل موردکاربرد نه به یکباره بلکه با طی چند گام انجام می‌شود که در هر گام بخشی از مدل موردکاربرد کامل می‌گردد. این کتاب، روشی سازمان یافته و گام به گام را برای این کار پیشنهاد و تشریح می‌کند.

روش‌های شناسایی کنشگرها، موردکاربردها، چگونگی تدوین قواعد کسب و کار و پیغام‌ها در موردکاربرد، شناسایی رابطه‌های تعیین، شمول و گسترش و مستندسازی آن‌ها در این فصل آمده است. در این فصل توصیه‌هایی برای هر مرحله پیشنهاد شده است که می‌تواند به عنوان چک‌لیست در بازنگری خروجی‌ها مورد استفاده قرار گیرد.

فصل هشتم: تدوین مشخصات تکمیلی

برخی از نیازمندی‌های نرم‌افزاری به شکل موردکاربرد قابل بیان نیستند. از آنجا که موارد کاربرد اکثر نیازمندی‌های سیستم را در خود جای داده است، این گونه نیازمندی‌ها تکمیل کننده مواردکاربرد هستند و از این رو آن‌ها را «تکمیلی» می‌نامیم و در سندي به نام «مشخصات تکمیلی» تدوین می‌کنیم. این فصل روش تدوین این گونه نیازمندی‌ها را تشریح می‌کند.

بخش پنجم: موضوعات تکمیلی

در این بخش دو موضوع تکمیلی مرتبط با نیازمندی‌های نرم‌افزاری شامل چالش‌های مهم در استخراج نیازمندی‌ها و تکنیک‌های آن ارائه شده است.

فصل نهم: چالش‌های استخراج نیازمندی‌ها

در این فصل چالش‌ها و آفت‌های استخراج نیازمندی‌ها معرفی شده و راه حل‌هایی برای آن‌ها ارائه شده است. ناآشنایی تیم توسعه با چالش‌های یادشده به مشکلات روابط انسانی دامن زده، آن‌ها را بغرنج‌تر می‌کند.

منبع اصلی این فصل:

Dean Leffingwell and Don Widrig, "Managing software requirements: A Use Case Approach", Addison-Wesley, 2003

فصل دهم: تکنیک‌های استخراج نیازمندی‌ها

تکنیک‌های استخراج نیازمندی‌ها بخشی از مهندسی نیازمندی‌ها است که تیم‌های توسعه بخصوص تحلیل‌گران از آن غافل می‌شوند. تحلیل‌گر با شناسایی شرایط، یک یا ترکیبی از چند تکنیک را برای استخراج نیازمندی‌ها به کار می‌گیرد. این فصل سعی دارد جعبه ابزاری از تکنیک‌ها برای خواننده فراهم کند.

بخش ششم: نمودارها

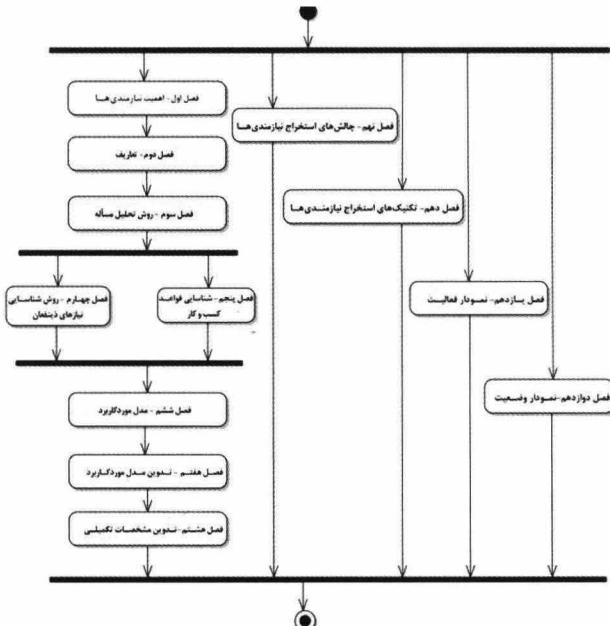
در این بخش دو نمودار از نمودارهای زبان UML- نمودار فعالیت و نمودار وضعیت- که در تحلیل نیازمندی‌ها کاربرد بیشتری دارند، معرفی و تشریح می‌شوند. این بخش شامل دو فصل است:

فصل یازدهم: نمودار فعالیت

فصل دوازدهم: نمودار وضعیت

روش مطالعه کتاب

می‌توان فصل‌های کتاب را به دو دسته مفهومی تقسیم نمود: فصل‌هایی که ضروری است به ترتیب مطالعه گردند (فصل‌های ۱ الی ۸) و فصل‌هایی که امکان مطالعه آن‌ها مستقل از سایر فصل‌ها وجود دارد (فصل‌های ۹، ۱۰، ۱۱ و ۱۲). دسته اول، فرایند پیشنهادی ارائه شده در این کتاب را دنبال می‌کند و دسته دوم، مطالب تکمیلی را در خود جای داده است. از این رو مسیر مطالعه پیشنهادی کتاب را می‌توان در قالب نمودار فعالیت به شکل زیر نمایش داد.



تصویر قبل در ابتدای تمامی فصل‌ها تکرار شده است تا علاوه بر یادآوری مسیر مطالعه به خواننده، مسیر اجرای فرایند تحلیل نیازمندی‌ها را نیز تصویرسازی نماید (یال سمت چپ). با وجود دسته‌بندی فصل‌های کتاب، پیشنهاد می‌گردد کتاب از آغاز تا پایان به صورت متواالی مطالعه گردد.

مثال‌های کتاب

در ارائه مثال‌ها سعی شده است از یک سیستم حساب سپرده کوتاه مدت - استفاده شود. در ادامه به تشریح این سیستم و پیش‌فرض‌های در نظر گرفته شده برای مثال‌های کتاب پرداخته می‌شود. بانک‌ها معمولاً دارای چهار نوع حساب سپرده ریالی برای مشتریان هستند: کوتاه‌مدت، بلند‌مدت، قرض الحسن و جاری.

فرض شده است که حساب سپرده کوتاه مدت تنها دارای فرایندهای افتتاح حساب، واریز به حساب، برداشت از حساب، مسدود کردن حساب، رفع مسدودی حساب، محاسبه واریز سود و بستن حساب است. برای سادگی، واریز به حساب و برداشت از آن تنها به شکل نقدی انجام می‌گردد و شکل‌های دیگر واریز و برداشت مانند برداشت و واریز با چک خارج از محدوده سیستم فرض شده است.

از طرف دیگر، فرض شده است که هر حساب تنها دارای یک صاحب حساب است و شکل‌های دیگر افتتاح حساب مانند قیم، وکیل و هم‌چنین حساب‌های مشترک خارج از محدوده فرض شده است. با این حال در هر مثال، سعی شده است فرض‌ها به شکلی مطرح گردند که به موضوع مرتبط باشند. ابزار مورد استفاده برای مدل‌سازی مثال‌های کتاب، Enterprise Architect یا اختصار EA، محصول شرکت Sparx است. برای مدل‌سازی می‌توان از ابزارهای دیگر که نمادهای UML 2.x را پشتیبانی کنند، استفاده کرد.

تقدیر و تشکر

نام بردن از همه عزیزانی که در طول یک و نیم سال تدوین کتاب و قبل از آن در جمع آوری و شکل گیری مطالب کتاب تأثیرگذار بوده‌اند، از حوصله کتاب خارج است. از همگی آن‌ها سپاسگزاریم و بهترین‌ها را برایشان آرزومندیم.

از استادان ارجمند آقایان دکتر رامان رامسین و مهندس نوید خسروی و دوستان عزیزمان جناب آقای مهندس علی‌رضا اسماعیلی و علی باقری صمیمانه تشکر می‌کنیم.

از نظرات و راهنمایی بسیاری از دوستان و همکاران بزرگوار در بهبود کیفیت کتاب بهره گرفته شده است که از تک‌تک آن‌ها از جمله مهندس مریم میرصالحی، مهندس ابوالفضل هادی، دکتر سمیه ملکوتی خواه، مهندس رضا بابانژاد، مهندس مسعود خاری، مهندس ریحانه جعفری و مهندس هوشنگ مختاری نیا قدردانی می‌کنیم.

برای جناب آقای مهندس مرتضوی- مدیرعامل شرکت پویا- و جناب آقای مهندس حدادی- مدیرعامل شرکت بريد سامانه نوين- بهترین‌ها را آرزومندیم.

از کمک‌های بی‌دریغ جناب آقای مهدی صادقی، مدیر محترم انتشارات رسم و مهندس علی‌رضا سلیمی در چاپ و انتشار این کتاب قدردانی می‌گردد.

هر چند که نه شایسته و نه قادر به قدردانی از استاد ارجمند جناب آقای ابراهیم نقیب‌زاده مشایخ هستیم، با این حال از زحمات و راهنمایی‌های ایشان بسیار سپاسگزاریم.

فهرست

۱۷	بخش اول: مقدمه.....	بخش اول: مقدمه.....
۱۸	فصل اول - اهمیت نیازمندی‌ها.....	فصل اول - اهمیت نیازمندی‌ها.....
۲۰	-۱ مقدمه.....	-۱ مقدمه.....
۲۰	پروژه موفق.....	-۲ پروژه موفق.....
۲۰	دلایل اصلی موفقیت یا شکست پروژه‌ها.....	-۳ دلایل اصلی موفقیت یا شکست پروژه‌ها.....
۲۲	میزان و اندازه رخداد خطاهای نیازمندی‌ها.....	-۴ میزان و اندازه رخداد خطاهای نیازمندی‌ها.....
۲۳	هزینه بالای خطاهای نیازمندی‌ها: قاعده ۱۰۰-۱	-۵ هزینه بالای خطاهای نیازمندی‌ها: قاعده ۱۰۰-۱
۲۶	نکات کلیدی.....	-۶ نکات کلیدی.....
۲۷	مراجع.....	-۷ مراجع.....
۲۸	فصل دوم - تعاریف.....	فصل دوم - تعاریف.....
۳۰	-۱ مقدمه.....	-۱ مقدمه.....
۳۰	نیازمندی.....	-۲ نیازمندی.....
۳۱	دسته‌بندی نیازمندی‌ها.....	-۳ دسته‌بندی نیازمندی‌ها.....
۳۵	سطوح نیازمندی‌ها.....	-۴ سطوح نیازمندی‌ها.....
۳۸	تفاوت نیازمندی و طراحی.....	-۵ تفاوت نیازمندی و طراحی.....
۴۰	قید.....	-۶ قید.....
۴۱	ذینفع.....	-۷ ذینفع.....
۴۳	مدیریت نیازمندی‌ها	-۸ مدیریت نیازمندی‌ها
۴۴	نکات کلیدی.....	-۹ نکات کلیدی.....
۴۵	مراجع.....	-۱۰ مراجع.....
۴۷	بخش دوم: تحلیل مسأله.....	بخش دوم: تحلیل مسأله.....
۴۸	فصل سوم - روش تحلیل مسأله.....	فصل سوم - روش تحلیل مسأله.....
۵۰	-۱ مقدمه.....	-۱ مقدمه.....
۵۰	-۲ مسأله چیست.....	-۲ مسأله چیست.....
۵۲	توافق بر سر تعریف مسأله.....	-۳ توافق بر سر تعریف مسأله.....
۵۳	شناسایی عوامل بروز مسأله.....	-۴ شناسایی عوامل بروز مسأله.....
۵۹	شناسایی ذینفعان و کاربران.....	-۵ شناسایی ذینفعان و کاربران.....
۶۴	تعیین مرز سیستم.....	-۶ تعیین مرز سیستم.....
۶۷	شناسایی قیدهای راه حل.....	-۷ شناسایی قیدهای راه حل.....
۶۸	نکات کلیدی.....	-۸ نکات کلیدی.....
۶۹	مراجع.....	-۹ مراجع.....
۷۱	بخش سوم: شناسایی نیازهای ذینفعان.....	بخش سوم: شناسایی نیازهای ذینفعان.....
۷۲	فصل چهارم - روش شناسایی نیازهای ذینفعان.....	فصل چهارم - روش شناسایی نیازهای ذینفعان.....
۷۴	-۱ مقدمه.....	-۱ مقدمه.....

۷۴	تشریح جایگاه محصول	-۲
۷۵	تدوین نیازهای اصلی ذینفعان و کاربران	-۳
۷۸	شناسایی ویژگی های سیستم	-۴
۸۳	توصیف کلان محصول	-۵
۸۵	شناسایی نیازمندی های مستندسازی	-۶
۸۵	شناسایی سایر نیازمندی های محصول	-۷
۸۶	سندها	-۸
۸۹	نکات کلیدی	-۹
۹۰	مراجع	-۱۰
۹۱	فصل پنجم - شناسایی قواعد کسب و کار	
۹۳	۱ - مقدمه	
۹۳	۲ - تعریف قواعد کسب و کار	
۹۴	۳ - اهمیت قواعد کسب و کار	
۹۵	۴ - خاستگاه عمومی قواعد کسب و کار	
۹۶	۵ - سطح بندي قواعد کسب و کار	
۹۷	۶ - طبقه بندي قواعد کسب و کار	
۹۹	۷ - شناسایی قواعد کسب و کار	
۹۹	۸ - تدوین قواعد کسب و کار	
۱۰۴	۹ - راهنمای تدوین قواعد کسب و کار	
۱۰۷	۱۰ - نکات کلیدی	
۱۰۸	۱۱ - مراجع	
۱۰۹	بخش چهارم: تعریف سیستم	
۱۱۰	فصل ششم - مدل مورد کاربرد	
۱۱۲	۱ - مقدمه	
۱۱۲	۲ - مدل مورد کاربرد	
۱۱۴	۳ - کنشگر	
۱۲۱	۴ - مورد کاربرد	
۱۲۳	۵ - سند مشخصات مورد کاربرد	
۱۳۰	۶ - بسته	
۱۳۰	۷ - سند مرور مدل مورد کاربرد	
۱۳۴	۸ - کاربرد مدل مورد کاربرد	
۱۳۶	۹ - نکات کلیدی	
۱۳۷	۱۰ - مراجع	
۱۳۸	فصل هفتم - تدوین مدل مورد کاربرد	
۱۴۰	۱ - مقدمه	
۱۴۰	۲ - شناسایی کنشگرها و تدوین مشخصات آنها	

۱۴۵	شناسایی موارد کاربرد و تدوین شرح مختصر آنها.....	-۳
۱۵۱	ترسیم نمودار مورد کاربرد اولیه.....	-۴
۱۵۱	طرح ریزی مشخصات مورد کاربرد.....	-۵
۱۵۴	تشریح مشخصات مورد کاربرد.....	-۶
۱۵۵	تقسیم موارد کاربرد.....	-۷
۱۶۰	شناسایی روابط بین موارد کاربرد	-۸
۱۶۶	استفاده از سایر نمودارها.....	-۹
۱۷۲	سازماندهی مدل مورد کاربرد.....	-۱۰
۱۷۵	توضیحات تکمیلی.....	-۱۱
۱۷۵	نکات کلیدی.....	-۱۲
۱۷۶	مراجع.....	-۱۳
۱۷۷	فصل هشتم - تدوین مشخصات تکمیلی.....	
۱۷۹	مقدمه.....	-۱
۱۷۹	جایگاه مشخصات تکمیلی.....	-۲
۱۸۰	شناسایی مشخصات تکمیلی.....	-۳
۱۸۶	قالب مستند مشخصات تکمیلی.....	-۴
۱۸۸	نکات کلیدی.....	-۵
۱۸۹	مراجع.....	-۶
۱۹۱	بخش پنجم: موضوعات تکمیلی.....	
۱۹۲	فصل نهم - چالشهای استخراج نیازمندی ها.....	
۱۹۴	مقدمه.....	-۱
۱۹۴	آفت «بله، اما».....	-۲
۱۹۵	آفت قلعه های کشف نشده.....	-۳
۱۹۶	آفت کاربر - توسعه دهنده.....	-۴
۱۹۸	نکات کلیدی.....	-۵
۱۹۹	مراجع.....	-۶
۲۰۰	فصل دهم - تکنیک های استخراج نیازمندی ها.....	
۲۰۲	مقدمه.....	-۱
۲۰۳	مصاحبه	-۲
۲۰۴	مشاهده	-۳
۲۰۴	تحقیق/پرسشنامه.....	-۴
۲۰۵	تحلیل مستندات.....	-۵
۲۰۶	مهندسی معکوس.....	-۶
۲۰۶	نمونه سازی.....	-۷
۲۰۷	توفان فکری.....	-۸
۲۰۸	گروه متمرکز.....	-۹

۲۰۹	شناسایی رابط	-۱۰
۲۰۹	نقالی	-۱۱
۲۱۰	ایفای نقش	-۱۲
۲۱۰	کارگاه نیازمندی‌ها	-۱۳
۲۱۱	انتخاب تکنیک مناسب	-۱۴
۲۱۳	نکات کلیدی	-۱۵
۲۱۴	مراجع	-۱۶
۲۱۵	بخش ششم: نمودارها	
۲۱۶	فصل یازدهم - نمودار فعالیت	
۲۱۸	- مقدمه	-۱
۲۱۹	- عناصر نمودار فعالیت	-۲
۲۲۷	نکات کلیدی	-۳
۲۳۸	مراجع	-۴
۲۳۹	فصل دوازدهم - نمودار وضعیت	
۲۴۱	- مقدمه	-۱
۲۴۲	«وضعیت»	-۲
۲۴۳	«گذار»	-۳
۲۴۶	«وضعیت مرکب»	-۴
۲۴۷	«وضعیت پایانی»	-۵
۲۴۷	«شبه وضعیت»	-۶
۲۵۱	نکات کلیدی	-۷
۲۵۲	مراجع	-۸
۲۵۳	پیوست‌ها	
۲۵۴	پیوست یک - واژه‌نامه	
۲۵۵	واژه‌نامه فارسی به انگلیسی	-۱
۲۶۱	واژه‌نامه انگلیسی به فارسی	-۲



بخش اول: مقدمه

در این بخش، ابتدا به بررسی اهمیت مرحله نیازمندی‌ها در توسعه سیستم‌های نرم‌افزاری پرداخته می‌شود و در ادامه، مجموعه‌ای از واژه‌های مهم این حوزه، تعریف می‌گردند.

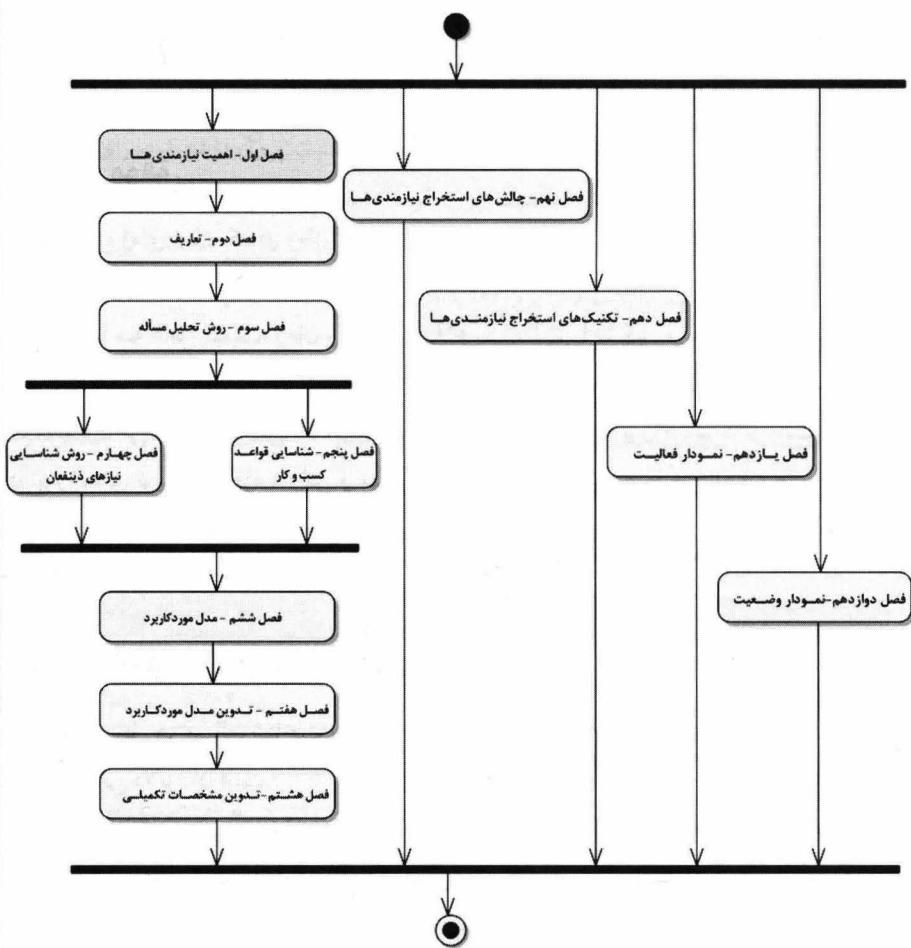
فصل اول – اهمیت نیازمندی‌ها

فصل دوم – تعاریف

فصل اول - اهمیت نیازمندی ها

اگر نیازمندی ها را به درستی شناسایی نکنید، خوب انجام دادن بقیه پروژه، دیگر اهمیتی نخواهد داشت.

کارل ای ویگرز نویسنده کتاب *More About Software Requirements*



۱- مقدمه

در این فصل سعی شده است با ارائه نتایج آماری و مطالعاتی، جایگاه و اهمیت نیازمندی‌ها در فرایند توسعه، استقرار و نگهداری سیستم‌های نرم‌افزاری تشریح گردد. در این راستا تعریفی از پروژه موفق ارائه شده است و در ادامه، جایگاه نیازمندی‌ها در موفقیت یا شکست پروژه‌ها بررسی شده است.

۲- پروژه موفق

پروژه موفق، پروژه‌ای است که در زمان تعیین شده، با بودجه پیش‌بینی شده و با سطح کیفی مورد نظر به انجام برسد.

در پروژه‌ها سه عامل کیفیت، زمان و هزینه، اضلاع یک مثلث را تشکیل می‌دهند که تغییر اندازه هر یک، موجب تغییر دو ضلع دیگر می‌گردد. برای مثال افزایش زمان منجر به افزایش هزینه پروژه و تغییر کیفیت محصول می‌گردد و افزایش سطح کیفیت، بر روی زمان و هزینه پروژه تأثیرگذار است. در پروژه‌های توسعه نرم‌افزار، اولین و مهمترین پارامتر کیفی، تأمین صحیح نیازمندی‌های مشتری است. در عین حال مدیریت خواسته‌ها و تأمین آن‌ها برای جلوگیری از تغییرات شدید دو پارامتر زمان و هزینه ضروری است.

۳- دلایل اصلی موفقیت یا شکست پروژه‌ها

اولین گام برای حل هر مسأله، شناخت ریشه و علت آن است. از این‌رو، کاهش میزان شکست پروژه‌ها، مستلزم ریشه‌یابی دلایل آن است.

مؤسسه استندیش گروپ^۱ تحقیقات وسیعی برای ریشه‌یابی علل شکست و موفقیت پروژه‌ها انجام داده است که با بررسی آمار حاصل از این تحقیقات می‌توان به اهمیت مدیریت نیازمندی‌ها در پروژه‌ها پی‌برد. بر اساس نتایج تحقیقات سال ۱۹۹۵ در این مؤسسه، شایع‌ترین مشکلات در توسعه نرم‌افزار با نیازمندی‌های آن ارتباط مستقیمی دارند. بر اساس این تحقیقات، مسائلی که عموماً پروژه‌های نرم‌افزاری را به چالش می‌کشند، به سه دسته تقسیم می‌گردند که در جدول زیر آورده شده است.

۱- مؤسسه آمریکایی Standish Group یکی از شرکت‌های پیشرو و مشهور در ارزیابی ریسک، هزینه، ارزش و بازگشت سرمایه در حوزه فناوری اطلاعات است. «ما روی شکست‌ها تمرکز می‌کنیم تا شما را در کسب موفقیت یاری کنیم» توصیف هدف این شرکت در وب‌گاه آن است. www.standishgroup.com

جدول ۱-۱: عوامل شکست پروژه‌های نرم‌افزاری

مسئله	درصد از کل پروژه‌ها
از دست رفتن و گم شدن اطلاعات و داده‌های ارائه شده توسط کاربر	% ۱۳
نیازمندی‌های ناقص و ناکافی	% ۱۲
تغییر نیازمندی‌ها	% ۱۲

گذشته از عوامل مذکور، سایر عوامل سهم کمتری در علل شکست پروژه‌ها دارند. برای مثال روشن است که زمان‌بندی غیر واقع‌بینانه، عامل شکست پروژه گردد. سهم این عامل در شکست پروژه‌ها ۴٪ است. نیروی انسانی ناکافی با ۶٪ و نداشتن مهارت‌های فناورانه با ۷٪؛ از دیگر عوامل شکست پروژه‌ها هستند. بنابر آمار استندیش گروپ، یک سوم از پروژه‌ها به دلیل اشکالات و تقایص جمع‌آوری، مستندسازی و مدیریت نیازمندی‌ها با چالش مواجه شده‌اند.

بر مبنای تحقیقی دیگر از استندیش گروپ، تنها ۹٪ از پروژه‌ها در شرکت‌های بزرگ در زمان و با بودجه مصوبشان به اتمام می‌رسند. این عدد برای شرکت‌های کوچک، ۱۶٪ است. سه عامل اصلی موفقیت پروژه‌ها بر اساس همین تحقیق در جدول زیر آورده شده‌اند.

جدول ۱-۲: عوامل موفقیت پروژه‌های نرم‌افزاری در سال ۱۹۹۵

عامل	درصد
مشارکت کاربران	در ۱۶٪ از پروژه‌های موفق
حمایت مدیران اجرایی	در ۱۴٪ از پروژه‌ها
بیان و اظهار واضح و بدون ابهام نیازمندی‌ها	در ۱۲٪ از پروژه‌ها

نتایج همین تحقیقات در سال ۲۰۰۶ نشان می‌دهد که سه عامل مؤثر در موفقیت پروژه‌ها عبارتند از:

- مشارکت کاربران
- حمایت مدیران اجرایی
- تعیین اهداف مشخص

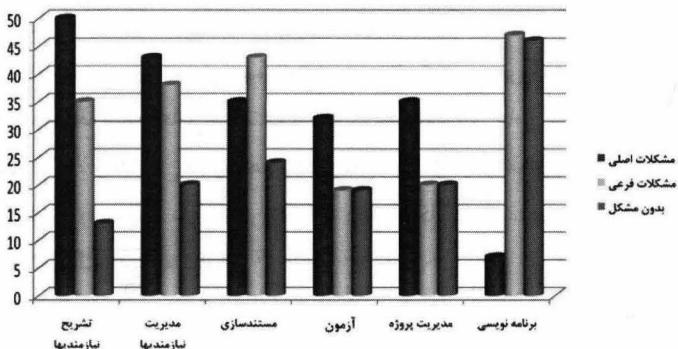
در این گزارش، وضعیت پروژه‌ها به شکل زیر ارائه شده است:

جدول ۱-۳: وضعیت پروژه‌ها در سال ۲۰۰۶

وضعیت پروژه	درصد
پروژه‌های موفق	% ۳۵
پروژه‌های ناموفق	% ۱۹
پروژه‌های مشکل‌دار	% ۴۶

بر اساس این نتایج درصد پروژه‌های موفق نسبت به سال ۱۹۹۵، رشد چشم‌گیری داشته است که

کماکان عامل «مشارکت کاربران» در صدر عوامل بهبود وضعیت پروژه‌ها قرار دارد. در سایر بررسی‌های انجام شده نیز نتایج مشابهی ارائه شده است. برای مثال ESPITI^۱ سرپرستی تحقیقی را بر عهده داشته است که در آن به بررسی و شناسایی دلایل مشکلات موجود در صنعت نرم‌افزار پرداخته شده است.^۲ نتیجه این بررسی که حاصل ۳۸۰۰ پاسخ‌نامه است در شکل ۱-۱ آمده است.



شکل ۱-۱: نمودار مشکلات اصلی توسعه نرم‌افزارها

این نمودار نشان می‌دهد که نزدیک به ۵۰٪ مشکلات بخش تشریح نیازمندی‌ها، مشکلات اصلی پروژه‌ها هستند. این در حالی است که تنها کمتر از ۱۰٪ مشکلات بخش کدنویسی، جزء این دسته از مشکلات بوده‌اند.

بر اساس نمودار، مشکلات بخش‌های «تشریح نیازمندی‌ها» و «مدیریت نیازمندی‌ها»، مشکلات اصلی توسعه نرم‌افزار بوده‌اند. با توجه به این که این بخش‌ها جزوی از حوزه نیازمندی‌ها هستند، می‌توان نتیجه گرفت که مشکلات این حوزه، جزو مشکلات اصلی پروژه‌های توسعه نرم‌افزار هستند. در ادامه اثرات اقتصادی این عامل بررسی می‌گردد.

۴- میزان و اندازه رخداد خطاهای نیازمندی‌ها

مطالعات دو نهاد استنديش گروپ و ESPITI داده‌هایی نه تنها کیفی، بلکه کمی را ارائه می‌دهند که نشان‌دهنده اثرات مخرب چالش‌های حوزه نیازمندی‌ها نسبت به سایر عوامل مخاطره‌زا در پروژه‌های نرم‌افزاری است. در اینجا پرسش مهم آن است که «با وجود اثرات مخرب چالش‌های حوزه نیازمندی‌ها بر توسعه سیستم‌های نرم‌افزاری، آیا چالش‌های مذکور اثرنامطلوبی بر روی محصول نهایی نیز خواهند داشت؟»

جدول ۱-۴: خلاصه‌ای از خطاهای بالقوه در توسعه سیستم‌های نرم‌افزاری را ارائه می‌دهد که حاصل

1. European Software Process Improvement Training Initiative

۲- داده‌ها مربوط به سال ۱۹۹۵ هستند.

مطالعات آقای کاپر جونز^۱ است. این مطالعات در زمینه خطاهای بالقوه و میزان موفقیت تیم‌های توسعه در رفع آن‌ها، در طی فرایندهای آزمون، بازنگری و سایر روش‌های کشف و رفع خطاست.

جدول ۱-۴: خلاصه‌ای از خطاهای بالقوه در توسعه سیستم‌های نرم‌افزاری

منشأ نقص و خطأ	تعداد نقص و خطأ	نسبت خطاهای رفع شده توسط تیم توسعه نسبت به کل خطاهای	تعداد خطاهای انتقال یافته به کاربران ^۲
نیازمندی‌ها	۱	%۷۷	۰.۲۳
طراحی	۱.۲۵	%۸۵	۰.۱۹
کدنویسی	۱.۷۵	%۹۵	۰.۰۹
مستندسازی	۰.۶۰	%۸۰	۰.۱۲
اصلاحات ناقص	۰.۴۰	%۷۰	۰.۱۲
مجموع	۵	%۸۵	۰.۷۵

داده‌های ستون «تعداد وقوع نقص و خطأ» از ۵ نرمال شده است و تعداد واقعی نقص و خطاهای را بیان نمی‌کند و ستون «تعداد خطاهای انتقال یافته به کاربران» نشان‌دهنده بخشی از خطاهاست که توسط کاربر مشاهده می‌شود. این مقدار نیز از ۵ نرمال شده است. به عنوان مثال از هر ۵ نقص و خطأ، منشأ یکی از آن‌ها، حوزه نیازمندی‌ها است و از هر خطایی که در این حوزه به وجود آمده است، ۰/۲۳ به کاربران منتقل شده است، در حالی که ۷۷٪ کل خطاهای این حوزه توسط تیم توسعه رفع شده است.

نتایج، نشان‌دهنده آن است که خطاهای نیازمندی‌ها بیشترین قسمت از خطاهای منتقل شده به کاربران را تشکیل می‌دهند، یعنی با وجودی که تعداد خطاهای کمتر از حوزه‌های طراحی و کدنویسی است، اما بالاترین میزان خطاهای انتقال یافته با ۰/۲۳ در اختیار این حوزه است. به عبارت دیگر، تقریباً یک سوم از کل خطاهای منتقل شده به کاربر (۰/۷۵ از ۰/۲۳)، خطاهای نیازمندی‌ها هستند.

۵- هزینه بالای خطاهای نیازمندی‌ها: قاعده ۱-۲۰۰

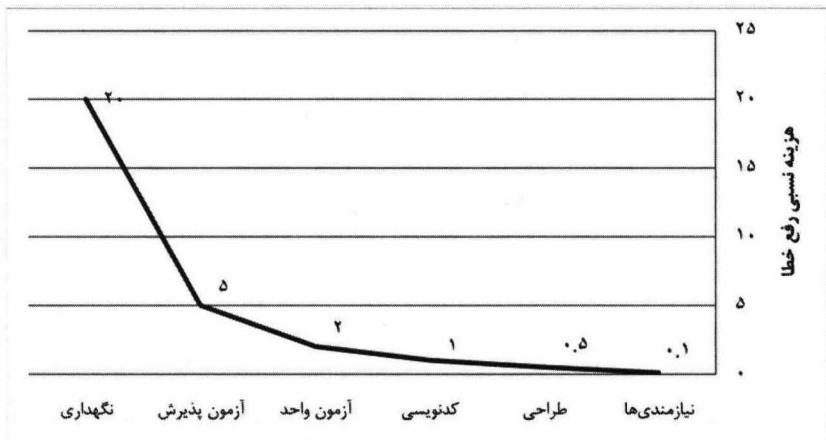
اگر خطاهای نیازمندی‌ها را بتوان به سرعت، به سادگی و به شکلی اقتصادی رفع نمود مشکل بزرگی وجود نخواهد داشت. اما آن چه اهمیت خطاهای و چالش‌های این حوزه را دو چندان می‌کند، هزینه‌های رفع

۱- کاپر جونز یکی از متخصصان متدولوژی‌های مهندسی نرم‌افزار است و بیشتر شهرتش به خاطر کارهایش در زمینه مدل نقطه کارکردی (function point) در برآورد هزینه است.

۲- تعداد خطاهای انتقال یافته به کاربران مساوی است با «تعداد نقص و خطأ» ضربدر (یک منهای «نسبت خطاهای رفع شده توسط تیم توسعه به کل خطاهای»)

دیرهنگام آن‌هاست.

شکل ۱-۲ براساس نتایج بررسی‌های انجام شده برای اندازه‌گیری هزینه خطاهای فازهای مختلف پروژه‌های نرم‌افزاری در شرکت‌های IBM، TRW و HP تهیه شده است. بر اساس این بررسی، اگر هزینه رفع خطایی در مرحله کدنویسی معادل یک واحد در نظر گرفته شود، هزینه رفع خطأ در مرحله نیازمندی‌ها، $0/1$ تا $0/2$ واحد و در مرحله نگهداری، 20 واحد خواهد بود.



شکل ۱-۲: نمودار نسبت هزینه رفع خطأ و نواقص در هر مرحله

برای محاسبه نسبت هزینه رفع خطایی که در مرحله نگهداری شناسایی شده است به هزینه رفع همان خطأ، در صورتی که در مرحله نیازمندی‌ها شناسایی می‌شد، کافی است تا عدد 20 به عددی بین $0/1$ تا $0/2$ تقسیم شود. این نسبت، عددی بین 100 تا 200 خواهد بود. به عبارت دیگر، نمودار مذکور بیانگر نسبت یک به دویست در هزینه کشف و رفع خطأ در مرحله نگهداری نسبت به همان عملیات در مرحله نیازمندی‌هاست. ممکن است این نسبت اغراق‌آمیز به نظر برسد، از این‌رو در ادامه سعی خواهد شد تا با رائمه عواملی که باعث تشدید و رشد غیرخطی هزینه‌ها می‌گردند، این نسبت توجیه گردد.

در شکل ۱-۲ برای هر مرحله، دو گونه هزینه، تجمعی و نتیجه حاصل ارائه شده است. این هزینه‌ها عبارتند از:

- هزینه خطاهای هر مرحله از تولید نرم‌افزار

- هزینه خطاهای منتقل شده از مراحل قبل

خطاهای هر مرحله به معنای خطاهای شناسایی شده در همان مرحله، بر اساس برنامه پروژه است. به عنوان مثال، خطاهایی که در مرحله طراحی کشف می‌شوند به دو دسته تقسیم می‌گردد:

- خطاهای ناشی از طراحی

اولین دسته خطاهای طراحی، خطاهایی هستند که با فرض وجود مجموعه‌ای صحیح و بدون اشکال از نیازمندی‌ها، در طراحی وجود دارند. این نوع خطاهای ناشی از فعالیت‌های طراحی هستند.

● خطاهای انتقالی از مرحله نیازمندی‌ها

دسته دوم خطاهای طراحی، خطاهایی را شامل می‌شوند که باید زودتر در مرحله نیازمندی‌ها کشف می‌شدند، ولی به دلایلی در مرحله نیازمندی‌ها کشف و رفع نشده و به مرحله طراحی منتقل شده‌اند. این نوع خطاهای اغلب بسیار پرهزینه‌اند، چرا که:

● خطاهای نیازمندی‌ها به مرور زمان کشف می‌شوند. در طی این زمان (بعد از اتمام مرحله نیازمندی‌ها و شروع مرحله طراحی و پیاده‌سازی)، گروه توسعه بر روی طراحی و پیاده‌سازی نیازمندی‌ها، زمان و هزینه صرف کرده است. بدیهی است بعد از کشف خطاهای انتقالی از مرحله نیازمندی‌ها و رفع آن‌ها در مستندات نیازمندی‌ها، فرایند تحلیل و طراحی باید دوباره انجام گردد.

● ممکن است خطا تغییر شکل یابد. از آنجا که تیم توسعه در آزمون و بازنگری مرحله طراحی، به دنبال خطاهایی از نوع خطاهای طراحی است، باید زمان و هزینه زیادی صرف گردد تا مشخص شود که خط، خطای طراحی نیست، بلکه خطایی از نوع خطاهای انتقالی مرحله نیازمندی‌ها است.

● مناسب‌ترین فرد برای کشف خطاهای نیازمندی‌ها، کاربران سیستم و یا تحلیل‌گران کسب‌وکار هستند که معمولاً در مرحله طراحی و پیاده‌سازی حضور کم‌رنگ‌تری دارند. اطمینان از درستی خطای کشف‌شده، نیازمند شناسایی کاربری است که نیازمندی را برای اولین بار مطرح کرده است. یافتن این فرد، کار آسانی نیست. از طرف دیگر امکان دارد که کاربر جزئیات را به خاطر نیاورد یا این که مشخص گردد وی دستورالعمل‌ها و مستندات لازم را در اختیار تیم توسعه قرار نداده است. برای انجام این کار نیاز به همراهی عضوی از تیم است که در آن مقطع، نقش «تحلیل‌گر کسب‌وکار» یا «تحلیل‌گر سیستم» را داشته است و احتمالاً اکنون در پروژه دیگری مشغول به کار است. موارد مذکور نیز بخشی از عوامل افزایش هزینه‌های رفع خطاهای نیازمندی‌ها در مراحل بعد هستند.

مشکل انتقال خطاهای به مرحله‌ای دیگر، موضوعی قابل کشف و شناسایی است. اما بسیاری از شرکت‌ها و سازمان‌ها، آن‌ها را با دقت و به درستی بررسی نمی‌کنند. در تحقیقاتی در شرکت «هیوز ایرکرافت» پدیده رسوخ و نشت ایرادها در مراحل مختلف پروژه‌های شرکت در طی ۱۵ سال بررسی شده است. این مطالعه نشان داده است که ۷۴٪ از نواقص و خطاهای نیازمندی‌ها در همان مرحله، کشف و رفع می‌گردد، مرحله‌ای که در آن مشتری و تحلیل‌گران سرگرم بحث، مذاکره و برگزاری جلسات برای شناسایی و مستندسازی نیازمندی‌ها هستند. این مرحله، ایده‌آل‌ترین زمان و فرصت برای کشف و رفع این نواقص است. این مطالعه در ادامه نشان می‌دهد که ۴٪ از خطاهای نیازمندی‌ها به مرحله طراحی کلان و ۷٪

به مرحله طراحی تفصیلی رسون می‌کنند و ۴٪ دیگر خطاهای زمان انتقال سیستم به محیط مشتری و نگهداری آن، شناسایی نمی‌شوند.

با توجه به زمان و مکان شناسایی خطاهای در فرایند توسعه نرم‌افزار، هزینه رفع خطا می‌تواند تا ۲۰۰ برابر افزایش یابد. دلایل افزایش تصاعدی هزینه مذکور شامل موارد زیر است:

- هزینه‌های بازطراحی، کدنویسی، آزمون، مستندسازی مجدد و اطلاع‌رسانی تغییرات به کاربران.
- هزینه‌های انجام عملیات اصلاحی برای جبران خسارت (به عنوان مثال عملیات تصحیح اطلاعات ثبت‌شده در پی وجود خطای که مانع از ورود اطلاعات نادرست نشده است).
- هزینه‌های ناشی از دور ریختن کدها، طراحی‌ها و آزمایه‌هایی^۱ که با دقت و وسواس زیاد اما مبتنی بر نیازمندی‌های اشتباہ تهیه شده‌اند.
- هزینه‌های فراخوانی محصولاتی که وجود اشکال در بخش نرم‌افزاری آن باعث عملکرد نادرست آن‌ها گردیده است. (روش متداول در صنعت خودرو)
- هزینه‌های ضمانت
- هزینه‌های مسئولیت، دیون و جریمه‌های احتمالی حاصل از عملکرد نادرست سیستم در پی شکایت مشتریان
- هزینه مراجعة به محل مشتری و انجام عملیات نصب و راهاندازی مجدد

۶- نکات کلیدی

- هدف از توسعه نرم‌افزار، تولید نرم‌افزارهایی با کیفیت، مطابق بودجه و زمان پیش‌بینی شده است که بتوانند نیازهای واقعی مشتریان را برآورده سازند.
- یکی از عوامل موفقیت پژوهه‌های توسعه نرم‌افزار، مدیریت کارآمد و مؤثر نیازمندی‌هاست.
- خطاهای نیازمندی‌ها متداول‌ترین و در عین حال پرهزینه‌ترین نوع خطا برای رفع شدن هستند.
- به کارگیری روش‌های استخراج و مدیریت نیازمندی‌ها می‌تواند منجر به کاهش قابل توجه خطاهای نیازمندی‌ها و افزایش کیفیت نرم‌افزار شود.

۷- مراجع

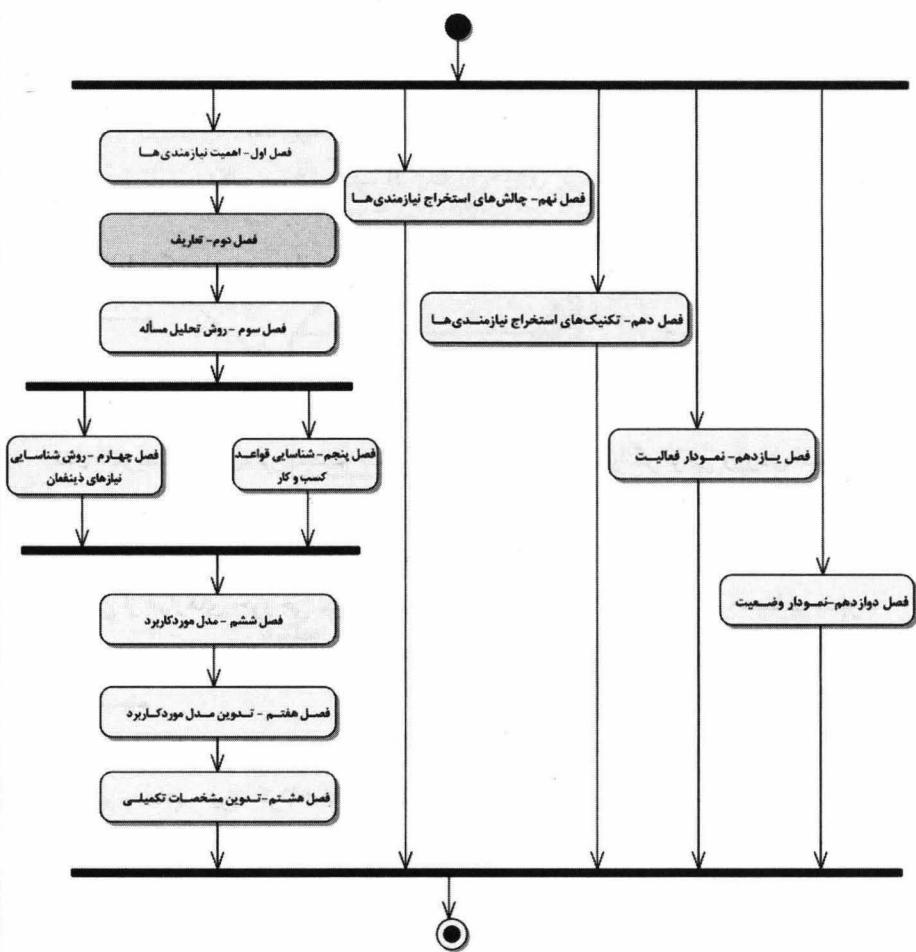
1. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
2. Leffingwell, Dean, Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
3. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
4. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>

فصل دوم - تعاریف

چهار کس را داد مردی یک درم هر یکی از شهری افتاده به هم فارسی و ترک و رومی و عرب جمله با هم در نزاع و در غصب فارسی گفتا ازین چون وارهیم هم بیا کاین را به انگوری دهیم آن یکی دیگر عرب بُد گفت لا من عنب خواهم نه انگور ای دغا آن یکی ترکی بُد و گفت ای گُزم آن یکی رومی به گفت این قیل را در تنازع مشت برهم میزند که ز سِرّ نامها غافل بُدند پُر بُدند از جهل و از دانش تهی صاحب سِرّی عزیزی صد زبان آرزوی جمله تان را می خرم پس به گفتی او که من زین یک درم مثنوی مولوی دفتر دوم

چون به معنی رفت آرام او فتاد
مثنوی مولوی دفتر دوم

اختلاف خلق از نام او فتاد



۱- مقدمه

واژگان تخصصی دارای معانی خاصی هستند که اگر به درستی استفاده نگردند، خطر انتقال نادرست مفاهیم و مطالب تشدید می‌شود. ممکن است در نوشتار و مکالمات عامه واژگان تخصصی به جای همدیگر به کار روند، اما در نوشتار و گفتار فنی بدین گونه نیست. از طرف دیگر واژگان تخصصی در منابع مختلف به تعابیر متفاوتی به کار برده شده‌اند. با توجه به مطالب مذکور در این بخش واژگان حوزه مدیریت نیازمندی‌ها تعریف می‌شوند.

۲- نیازمندی

برای معرفی کامل یک سیستم نرم‌افزاری لازم است پنج مؤلفه زیر توصیف شوند (Davis, 1993) ورودی سیستم

برای توصیف ورودی‌ها علاوه بر شرح داده‌های ورودی، در صورت نیاز، جزئیات دستگاه‌ها، شکل و شمایل داده‌ها و قراردادهای^۱ ورودی نیز توصیف می‌گردد. به عنوان مثال «تعریف مشتری» در سیستم حساب سپرده کوتاه مدت، بیانگر نوعی ورودی است.

خروجی سیستم

برای توصیف خروجی‌ها، علاوه بر داده‌های تولید شده سیستم، شکل و قرارداد آن‌ها، شرحی از ابزارهای خروجی نیز ارائه می‌گردد. به عنوان مثال در سیستم حساب سپرده کوتاه مدت «چاپ سند پس از ثبت عملیات واریز» نمونه‌ای از خروجی است.

وظیفه^۲ سیستم

برای شرح وظیفه‌ها، لازم است نحوه نگاشت ورودی‌ها به خروجی‌ها ارائه گردد. در سیستم حساب سپرده کوتاه مدت که یکی از وظایف آن، «واریز پول به حساب مشتری» است، شماره حساب و مبلغ به عنوان ورودی دریافت می‌شود و موجودی حساب افزایش می‌یابد.

خصوصیات^۳ سیستم

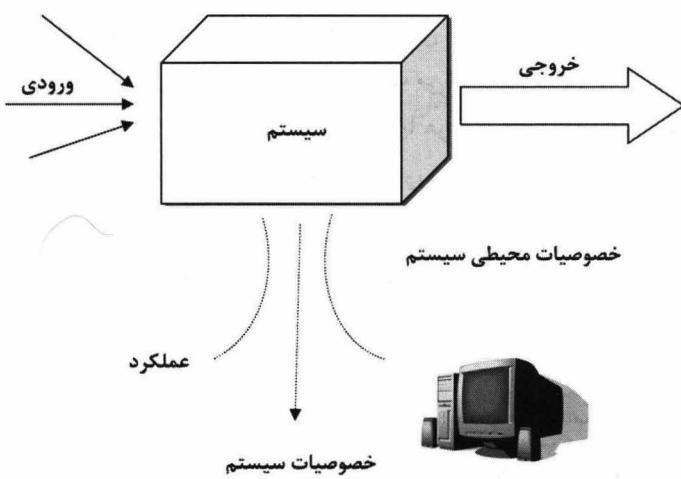
خصوصیات غیررفتاری سیستم شامل مواردی مانند قابلیت اطمینان^۴، قابلیت نگهداری^۵، دسترس پذیری^۶ و توان عملیاتی^۷ است که در توسعه سیستم باید مد نظر قرار گیرند. به

-
1. Protocols
 2. Functionality
 3. Attributes
 4. Reliability
 5. Maintainability
 6. Availability
 7. Throughput

عنوان مثال «امکان کار هزار متعدد امور بانکی در هر لحظه با سیستم» از خصوصیات سیستم حساب سپرده کوتاه مدت است.

شرایط محیطی سیستم

شرایط محیطی سیستم بیانگر قابلیت‌های غیررفتاری سیستم، مرتبط با محیط اجرای آن است. از جمله این قابلیت‌ها می‌توان به تعامل با سایر سیستم‌ها یا سیستم عامل‌های مختلف اشاره کرد. این دسته از قابلیت‌ها، در حقیقت بخش تکمیلی خصوصیات سیستم و بیانگر شرایطی هستند که سیستم به ناچار باید خود را با آن شرایط تطبیق دهد. در سیستم حساب سپرده کوتاه مدت «اجرای سیستم بر روی مرورگر IE نسخه ۶ به بالا» بیانگر شرایط محیطی آن است.



قابلیت اطمینان، کارایی، توان عملیاتی و...

شکل ۱-۲: سیستم و اجزای آن

مولفه‌های مذکور بیانگر قابلیت‌هایی هستند که در سیستم وجود دارد یا به شرایطی اشاره می‌کند که سیستم تحت آن شرایط کار می‌کند، برای پوشش دو مفهوم مذکور-قابلیت و شرایط - واژه نیازمندی به شکل زیر تعریف می‌گردد:

«نیازمندی نرم‌افزاری شرط یا قابلیتی است که سیستم باید آن را پوشش دهد.»

-۳- دسته‌بندی نیازمندی‌ها

یکی از اهداف طبقه‌بندی در علوم، شناخت بهتر پدیده‌ها است. به عنوان مثال وقتی که موجودات به دو دسته جاندار و بی‌جان تقسیم می‌گردند، هدف کاهش دشواری شناخت و شناسایی کامل‌تر

موجودات است. به عنوان نمونه اگر موجودی جاندار باشد، برای شناخت کامل وی ضروری است محل زندگی، طول عمر و روش زندگی وی، بررسی و تحلیل گردد. در حالی که پرسش‌های مذکور برای موجودات بی جان بی معنی خواهد بود.

در ادامه برای شناخت بهتر و استخراج اطلاعات جامع‌تر از نیازمندی‌ها، روش‌های دسته‌بندی آن‌ها ارائه می‌شود.

۱-۳ - دسته‌بندی کلی نیازمندی‌ها

به طور کلی نیازمندی‌ها به دو دسته نیازمندی‌های کارکردی^۱ و غیرکارکردی^۲ تقسیم می‌شوند. نیازمندی‌های کارکردی، کارهایی هستند که سیستم باید بدون در نظر گرفتن محدودیت‌ها قادر به انجام آن‌ها باشد. به عبارت ساده‌تر، نیازمندی‌های کارکردی بیانگر رفتارها و عملکرد سیستم هستند. این نیازمندی‌ها وظایف سیستم یا یکی از اجزای آن را معرفی می‌کنند. هر وظیفه از سیستم به شکل مجموعه‌ای از ورودی‌ها، رفتار و خروجی‌های آن توصیف می‌شود. نیازمندی کارکردی ممکن است کارهای محاسباتی، ثبت و تغییر داده‌ها، تولید خروجی‌ها یا هر عملکرد مورد انتظار دیگری از سیستم باشد. برای مثال در سیستم حساب سپرده کوتاه مدت «واریز به حساب»، «محاسبه و واریز سود حساب»، «چاپ گردش حساب» و «افتتاح حساب» نیازمندی‌های کارکردی هستند.

نیازمندی غیرکارکردی محدودیت یا شرایطی را برای عملکرد سیستم تعیین می‌کند. همان‌طور که گفته شد، نیازمندی‌های کارکردی، عملکرد سیستم را مشخص می‌کنند، درحالی که نیازمندی‌های غیرکارکردی، معیار پذیرش عملکرد را توصیف می‌نمایند. نیازمندی‌های غیرکارکردی محدودیت‌ها و معیارهای کیفی هستند که بر رفتار سیستم اعمال می‌گردد. به عنوان مثال امکان «واریز به حساب مشتری» عملکرد مورد انتظار از سیستم حساب سپرده کوتاه مدت، یک نیازمندی کارکردی است. در حالی که «عملیات واریز توسط هزار کاربر همزمان بیش از ده ثانیه طول نکشد» شرایط و محدودیت حاکم بر این عملکرد را بیان می‌نماید-نیازمندی غیرکارکردی. هر چند «واریز به حساب مشتری» باید به درستی انجام شود، اما این موضوع برای پذیرش آن کافی نیست، بلکه اطمینان از انجام آن مطابق شرایط ذکر شده (زمان کمتر از ده ثانیه برای هزار کاربر همزمان) شرط پذیرش آن است.

نیازمندی‌های غیرکارکردی «خصوصیات کیفی^۳» یا «اهداف کیفی^۴» نیز نامیده می‌شوند.

۲-۳ - مدل FURPS+

در اینجا دسته‌بندی دیگری از نیازمندی‌ها مورد بررسی قرار می‌گیرد که در آن، نیازمندی‌ها بر اساس

1. Functional
2. Non-Functional
3. Quality attributes
4. Quality goals
5. Functionality, Usability, Reliability, Performance, Supportability, + (others)

ماهیت، دسته‌بندی شده‌اند. این دسته‌بندی FURPS+ نامیده می‌شود. FURPS+ سرنامی^۱ مشکل از عنوان دسته‌های اصلی نیازمندی‌ها است. دسته‌های یاد شده عبارتند از:

۱. نیازمندی‌های کارکردنی

نیازمندی‌های کارکردنی به نیازمندی‌هایی اشاره دارند که قابلیت‌های عملکردی سیستم را مشخص می‌کنند.

۲. خوش‌کاری^۲

این دسته از نیازمندی‌ها، بیانگر قابلیت‌هایی مانند یکپارچگی رابط کاربر و راحتی کار با سیستم است. فهرست زیر نمونه‌هایی از این دسته از نیازمندی‌ها هستند:

- عوامل انسانی
- زیبایی
- همخوانی رابط کاربر
- وجود راهنمای لحظه‌ای و حساس به موضوع
- وجود ویژاردها^۳ و عامل‌ها^۴
- وجود مستندات کاربر
- وجود مستندات آموزشی

۳. قابلیت اطمینان^۵

این دسته از نیازمندی‌ها، مدت زمان در دسترس بودن سیستم و تعداد خطاهای قابل پذیرش آن را بیان می‌کند. فهرست زیر نمونه‌هایی از این دسته از نیازمندی‌ها است:

- تکرار و شدت عیوب و نقایص
- ترمیم و بازیابی خطاهای
- دقت و صحت کارکرد سیستم
- متوسط زمان بین کشف عیوب و نقایص و رفع آن‌ها

۴. کارایی^۶

کارایی «میزان کار در واحد زمان» است، نمونه‌هایی از این دسته نیازمندی‌ها، در زیر آمده است:

- زمان پاسخ‌گویی(میزان کار قابل انجام در واحد زمان)
- سرعت
- اثربخشی
- توان عملیاتی

1. Acronym

2. Usability

3. Wizards

4. Agents

5. Reliability

6. Performance

- زمان ترمیم و بازیابی
- استفاده از منابع (مانند حافظه، پردازنده و دیسک سخت)
- ۵. قابلیت پشتیبانی^۱

این گروه از نیازمندی‌ها حوزه پشتیبانی سیستم را در بر می‌گیرد. از جمله این نیازمندی‌ها می‌توان

به موارد زیر اشاره کرد:

- آزمون‌پذیری
- توسعه پذیری
- انطباق‌پذیری
- قابلیت نگهداری
- سازگاری
- قابلیت پیکربندی
- قابلیت تعمیر
- قابلیت نصب
- قابلیت محلی‌سازی

+ ۶.

علامت + نشان‌دهنده سایر خصوصیات کیفی است که در ادامه معرفی شده‌اند:

﴿ نیازمندی‌های (قیدهای) طراحی

نیازمندی‌های طراحی که غالباً قیدهای طراحی نیز نامیده می‌شوند، بیانگر گزینه‌های قابل انتخاب در طراحی یا محدودیت‌های طراحی سیستم هستند. به عنوان مثال در سیستم حساب سپرده کوتاه مدت، لزوم رعایت استانداردهای واحد فناوری اطلاعات بانک، استفاده از SQL Server را به عنوان بانک اطلاعاتی، به شکل یک نیاز طراحی بر پروژه تحمیل می‌کند.

﴿ نیازمندی‌های (قیدهای) پیاده‌سازی

نیازمندی‌های پیاده‌سازی بیانگر محدودیت‌ها یا گزینه‌های قابل انتخاب در کدنویسی و ساخت سیستم هستند. از جمله:

- استانداردهای توسعه
- زبان‌های پیاده‌سازی
- خط‌مشی‌های یکپارچگی بانک داده‌ها
- محدودیت منابع

○ محیط‌های عملیاتی

﴿ نیازمندی‌های (قیدهای) رابط

این گروه نیازمندی‌ها، موضوعات مرتبط با رابط ارتباطی سیستم را مشخص می‌کنند. برای نمونه می‌توان به موارد زیر اشاره نمود:

- ارتباط با عناصر خارجی مانند سیستم‌ها و سرویس‌هایی که تعامل با آن‌ها الزامی است.
- قیدهای موجود در تعامل با عناصر خارجی مانند شکل ارتباط، زمان و نحوه دسترسی، داده‌های ارسالی و دریافتی.

﴿ نیازمندی‌های (قیدهای) فیزیکی

این گروه از نیازمندی‌ها مشخصات اجسام فیزیکی مرتبط با سیستم را بیان می‌کنند. برای مثال مشخصات سخت‌افزارهای مورد نیاز برای راهاندازی و استقرار در این گروه توصیف می‌گردند. در فهرست زیر نمونه‌هایی از این نیازمندی‌ها آورده شده است:

- مواد به کار رفته^۱
- شکل
- اندازه
- وزن

۴- سطوح نیازمندی‌ها

هدف از سطح‌بندی موضوعات در یک حوزه دانش، مدیریت پیچیدگی‌ها و جزئیات آن است. در این روش، جزئیات از سطوح بالاتر به سطوح پایین‌تر افزایش می‌یابند. معمولاً سطوح بالاتر دارای اهمیت بیشتر با تعداد اعضای کمتر و سطوح پایین‌تر دارای اهمیت کمتر با تعداد اعضای بیشتری هستند. در اینجا نیز هدف از سطح‌بندی نیازمندی‌ها، مدیریت جزئیات و پیچیدگی‌ها بر مبنای اهمیتشان است. از این پس، هر سطح از نیازمندی‌ها «نوع» نامیده می‌شود. روش‌های مختلفی برای سطح‌بندی نیازمندی‌ها وجود دارد. در این کتاب روشی سه سطحی ارائه می‌گردد.

۱- به عنوان مثال بارکدخوان نمونه‌ای از وسایل ساخت‌افزاری برای راهاندازی سیستم است که ممکن است اندازه، شکل و وزن آن برای استفاده در فروشگاه‌ها، حائز اهمیت باشد. اگر فروشگاه در منطقه آب و هوایی خاصی قرار داشته باشد (مثلاً شمال کشور که رطوبت آن بالاست)، بارکدخوانی باید انتخاب گردد که مواد تشکیل‌دهنده آن با شرایط مرطوب سازگار باشند.



شکل ۲-۲: سطوح نیازمندی‌ها

مطابق شکل ۲-۲، انواع نیازمندی‌ها عبارتند از:

- ﴿ نوع اول: نیازهای^۱ ذینفعان^۲ ﴾
- ﴿ نوع دوم: ویژگی‌ها^۳ ﴾
- ﴿ نوع سوم: نیازمندی‌های نرم‌افزاری^۴ ﴾

نیاز

مهم‌ترین بخش حل هر مسأله در توسعه نرم‌افزار، چگونگی رفع نیازهای ذینفعان است. عموماً ذینفعان دیدگاه‌های متفاوت و مختلفی نسبت به مسأله دارند و سیستم ناگزیر باید نیازهای آن‌ها را برآورده سازد. «بازتاب مشکلات یا فرصت‌های کسب‌وکار که منجر به توجیه اهمیت، خرید یا استفاده از سیستم جدید می‌گردد»، نیاز نامیده می‌شود. به عنوان مثال «امکان خدمت‌رسانی به مشتری در تمامی شعب علاوه بر شعبه افتتاح‌کننده حساب» یکی از نیازهای ذینفعان سیستم حساب سپرده کوتاه مدت است، چرا که آنان انتظار دارند خدمات بهتری به مشتریان ارائه دهند و مشتریان بتوانند با مراجعته به هر شعبه‌ای از بانک، امکان استفاده از خدمات بانکی مانند واریز، برداشت، دریافت گزارش گردش حساب و اطلاع از موجودی حساب را داشته باشند. رفع نیازها مستلزم شناسایی، ثبت و پیگیری نیازهای ذینفعان در فرآیند توسعه است. در این بین شناسایی نیازها از اهمیت بسزایی برخوردار است. تیم توسعه برای استخراج و ارائه نیازهای ذینفعان نیازمند استفاده از شیوه‌ای مؤثر است تا مانع از بیان بی‌قاعده و غیررسمی^۵ آن‌ها گردد. در فصل‌های بعد روش‌هایی برای شناسایی نیازها ارائه می‌گردد.

در شکل ۳-۲ نشان داده شده است که نیازها در دامنه مسأله وجود دارند و از این‌رو با ادبیات دامنه مسأله مطرح می‌گردند. نیازها روشنی برای حل مسأله یا استفاده از فرصت ارائه نمی‌کنند، بلکه تنها بیان مشکل به زبان ذینفعان هستند.

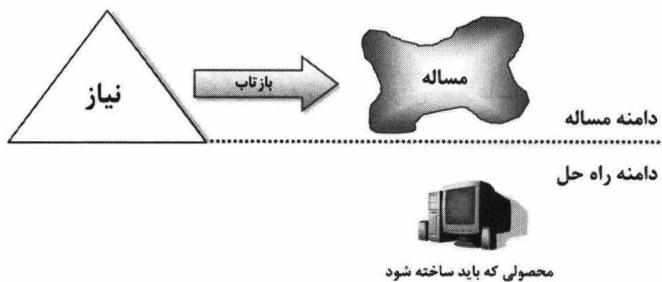
1. Need

۲- مفهوم ذینفع در همین فصل، توضیح داده می‌شود.

3. Feature

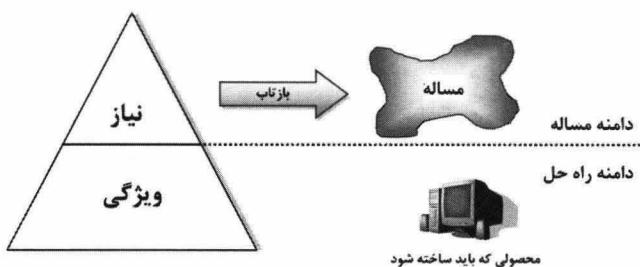
4. Software Requirement

5. Informal



شکل-۲-۳: جایگاه نیاز در دامنه مسأله

ویژگی‌ها



شکل-۲-۴: جایگاه ویژگی‌ها در راه حل

همان طور که در شکل ۲-۴ مشخص شده است، ویژگی‌ها در سطحی پایین‌تر از نیازها و در فضای راه حل قرار گرفته‌اند. به عبارت ساده‌تر، ویژگی‌ها قابلیت‌های کلان سیستم برای ارائه ارزش افزوده به ذینفع به منظور رفع نیازهای وی هستند.

از ویژگی‌ها می‌توان برای برنامه‌ریزی پروژه نیز استفاده کرد. به نمونه‌های زیر توجه کنید:

ویژگی "... در نسخه بعدی پیاده‌سازی خواهد شد.

ویژگی "... باید قبل از سایر ویژگی‌ها پیاده‌سازی شود.

ویژگی "... که توسط مشتری اعلام شده است، اما قابل قبول نیست.

ویژگی "... نیاز به بررسی بیشتر دارد.

ویژگی "... برای کاربر بسیار با اهمیت است.

ویژگی‌ها اطلاعات کافی را برای کارهای توسعه سیستم شامل طراحی، کدنویسی و آزمون سیستم ارائه نمی‌دهند، زیرا دارای تعریفی دقیق یا سطح یک‌دستی از نیازمندی‌ها نیستند و نمی‌توانند مستقیماً برای توسعه سیستم استفاده گردند. از این‌رو به مفاهیم دیگری برای تشریح جزئیات قابلیت‌های سیستم نیاز است که «نیازمندی نرم‌افزار» نامیده می‌شود.

نیازمندی‌های نرم‌افزار

همانطور که در بخش قبل بیان شد، ویژگی‌ها به تنها ای دارای اطلاعات کافی برای تعیین محدوده یا عملکرد سیستم نیستند و به علت کلی گویی نمی‌توانند ورودی‌های کامل، واضح و دقیقی برای مرحله طراحی و کدنویسی باشند. ویژگی‌ها تنها قابلیت‌های موجود در سیستم را فهرست‌وار به همراه چرایی، بدون ذکر چگونگی پوشش آن‌ها بیان می‌کنند.

برای تکمیل این اطلاعات ضروری است جزئیات بیشتر و دقیق‌تری از راه حل در قالب «نیازمندی‌های نرم‌افزاری» ارائه گردد. «نیازمندی نرم‌افزاری، مشخصات رفتار خارجی سیستم مانند ورودی‌ها، خروجی‌ها، کارکردها، خصوصیات سیستم یا خصوصیات محیطی آن است.»

به عنوان مثال به نیازمندی‌های نرم‌افزاری زیر توجه کنید:

○ نیازمندی نرم‌افزاری ۱: در هنگام ثبت مشخصات مشتری، پس از ورود کدم‌لی، در صورتی که اطلاعات مشتری در سیستم وجود داشته باشد، سیستم پیغام «اطلاعات مشتری قبل‌ثبت شده است» را به کاربر نمایش داده و از ادامه کار جلوگیری می‌کند.

○ نیازمندی نرم‌افزاری ۲: هنگام ثبت مشتری پس از انتخاب گرینه تأیید توسط کاربر، سیستم اطلاعات مشتری را ثبت می‌کند و سپس کد مشتری را صادر می‌نماید و به کاربر نمایش می‌دهد.

○ نیازمندی نرم‌افزاری ۳: در هنگام انجام عملیات واریز پول به حساب مشتری، سیستم اطلاعات کاربر جاری را به عنوان کاربر انجام‌دهنده عملیات واریز به همراه سایر اطلاعات ثبت می‌کند.

مثال‌های بالا نشان می‌دهند که نیازمندی‌های نرم‌افزاری دارای جزئیات کافی برای انجام فعالیت‌هایی مانند طراحی، کدنویسی و آزمون سیستم هستند.

نیازمندی‌های نرم‌افزاری در قالب مستندات «مشخصات مورد کاربرد^۱» و «مشخصات تکمیلی^۲» تدوین می‌گردند.^۳

۵- تفاوت نیازمندی و طراحی

در همین فصل بیان شد که برخلاف ویژگی‌ها که قابلیت و توانایی سیستم و چرایی آن‌ها را به شکل کلان توصیف می‌کنند، نیازمندی‌های نرم‌افزاری چگونگی تحقق آن‌ها را تشریح می‌نمایند. در این

1. Use case Specification

2. Supplementary Specification

۳- این مستندات در فصل‌های آتی به تفصیل شرح داده خواهد شد. به طور خلاصه می‌توان گفت که مورد کاربرد شرح تعامل سیستم با کاربران، سیستم‌های دیگر و ساخت‌افزارهای مرتبط با آن است و مشخصات تکمیلی حاوی نیازمندی‌های غیر کارکردی یا کارکردی مربوط به کل سیستم است.

تعریف توجه به کلمه «چگونگی» حائز اهمیت است. منظور از چگونگی، تشریح طراحی و کدنویسی نیست، بلکه بیان جزئیات و کیفیت رفتار سیستم است، به گونه‌ای که توسط کاربر و ذینفع قابل فهم باشد. از همین راست که نیازمندی‌ها به زبان کاربر تدوین می‌گردد. برای روشن شدن موضوع به مثال زیر توجه نمایید:

ویژگی:

○ امکان برداشت پول از حساب مشتری

نیازمندی نرم‌افزاری:

○ کاربر شماره حساب مشتری را وارد می‌کند.

○ سیستم اطلاعات حساب شامل موجودی و مشخصات صاحب حساب شامل نام، نام

خانوادگی، شماره ملی، نام پدر، شماره شناسنامه و تصویر امضاء وی را نمایش می‌دهد.

○ کاربر مبلغ برداشت را وارد می‌کند.

○ کاربر گزینه‌ی تأیید را انتخاب می‌کند.

.....

در نیازمندی نرم‌افزاری مذکور شرحی از چگونگی «برداشت پول از حساب مشتری» به زبان کاربر آورده شده است. در اینجا اثری از فناوری، بانک اطلاعاتی و تکنیک‌های طراحی نرم‌افزار وجود ندارد. حال به متن زیر توجه نمایید:

○ برای دریافت اطلاعات از بانک اطلاعات، دیدگاهی^۱ به نام VWCustomer می‌سازیم که اطلاعات مورد نظر را از جداول متناظر استخراج کند.

○ برای ثبت اطلاعات برداشت، رویه ذخیره‌شده‌ای^۲ به نام SPWithdraw نوشته می‌شود که با گرفتن پaramترها، اطلاعات واریز را در جدول TBLWithdraw ثبت می‌کند و مقدار ستون Balance (موجودی) در جدول TBLAccount را تغییر می‌دهد.

○ در صورتی که عملیات با موفقیت همراه نبود، WithdrawException به فرم ارسال می‌گردد و فرم نیز به کاربر پیغام می‌دهد. پیغام‌ها در Resource Bundle نگهداری می‌شوند و تنها کد آن‌ها در Exception‌ها قرار می‌گیرد.

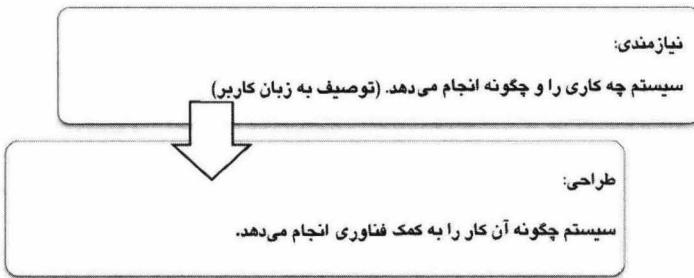
...

متن بالا نمونه‌ای از توصیف طراحی نرم‌افزار است^۳ که چگونگی ساخت محصول مبنی بر نیازمندی‌ها را با امکانات فناوری از جمله بانک‌های اطلاعاتی تشریح نموده است و دارای تفاوت‌های فاحشی با نیازمندی‌های نرم‌افزاری است. بنابراین منظور از «چگونگی»، بیان جزئیات بیشتر و دقیق‌تری از رفتار و کیفیت رفتار سیستم به زبان کاربر است.

1. View

2. Stored Procedure

3. Design Specification



شکل ۲-۵: تفاوت نیازمندی و طراحی

۶- قید

با شناسایی نیازهای ذینفعان تیم توسعه به دنبال ارائه راه حلی برای رفع آنها خواهد بود. اما ارائه راه حل همواره با محدودیت‌هایی همراه است که آزادی عمل تیم توسعه را تحت تأثیر قرار می‌دهد. به عنوان مثال در سیستم حساب سپرده کوتاه مدت ذینفع انتظار دارد که امکان افتتاح حساب برای مشتریان در سیستم وجود داشته باشد. تیم توسعه می‌تواند راه حلی به شکل زیر ارائه دهد:

ابتدا مشخصات مشتریان دریافت شود و شماره مشتری به آنها اختصاص یابد و در ادامه در فرم افتتاح حساب از شماره مشتری برای افتتاح استفاده گردد. برای مراجعتی که شماره مشتری دارند نیازی به ثبت مجدد اطلاعات و تخصیص شماره مشتری نخواهد بود.

حال فرض کنید کارفرما اعلام می‌نماید که اطلاعات مشتریان و تخصیص شماره به آنها در یکی از سیستم‌های موجود بانک به نام سیستم «اطلاعات مشتریان» انجام می‌شود، در نتیجه راه حل قبلی تیم توسعه معتبر نخواهد بود. آنها مجبورند راه حلی ارائه دهند که با شرایط اعلام شده مطابقت داشته باشد. این گونه شرایط «قید» نامیده می‌شوند که به شکل زیر تعریف می‌شوند.

«قید محدودیت یا شرطی برای ارائه راه حل است که آزادی عمل تیم توسعه را کاهش می‌دهد.» نمونه‌هایی از قیدها در جدول ۱-۲ آورده شده‌اند.

جدول ۱-۱: نمونه قید

دلیل	قید
نیاز به کترول و کاهش هزینه خرید و نگهداری سرویس‌دهنده‌ها	سیستم جدید بر روی سرویس‌دهنده‌های موجود نصب و راهاندازی خواهد شد.
سال مالی بانک آخر اسفند پایان می‌یابد و کلیه امور شعب در سال مالی جدید با سیستم انجام گردد.	سیستم باید قبل از فروردین ماه سال آینده راهاندازی شده باشد.
الزام به رعایت مصوبه کمیته راهبردی فناوری اطلاعات بانک در حوزه فناوری سیستم‌ها	سیستم مبتنی بر فناوری .Net. و بانک اطلاعاتی SQL Server 2008 باشد.

برخی از قیدها منجر به تعریف نیازمندی‌هایی در سیستم و برخی منجر به تغییر منابع، طرح‌ها و زمان‌بندی پروژه می‌شوند. به عنوان مثال قید «لزوم ارتباط سیستم با سیستم اطلاعات مشتریان که حاوی اطلاعات مشتریان فعلی بانک است» منجر به تعریف مجموعه‌ای از نیازمندی‌ها برای ارتباط و استفاده از داده‌های سیستم مذکور می‌گردد و قید «لزوم رعایت استانداردهای توسعه نرم‌افزار واحد فناوری اطلاعات» منجر به انتخاب SQL Server به عنوان بانک اطلاعاتی سیستم خواهد گردید.

۷- ذینفع^۱

همانطور که گفته شد نیازها رکن اساسی در توسعه سیستم هستند، چرا که سیستم برای رفع آن‌ها ساخته می‌شود. ذینفعان کسانی هستند که نیازها را در پروژه اعلام می‌نمایند. با توجه به اهمیت ذینفعان، یکی از موضوعات مهم در مدیریت نیازمندی‌ها، شناسایی ذینفعان پروژه است. ذینفعان اشخاص یا گروه‌هایی هستند که برای دست‌یابی به نتیجه مطلوب ضروری است نیازها و انتظاراتشان در پروژه برآورده گردد.

ممکن است ذینفع، کاربر سیستم نباشد. برای مثال به نمونه‌های زیر توجه کنید:

نمونه ۱: در پروژه توسعه سیستم آژانس هواپیمایی که تنها کارکنان آژانس با آن کار می‌کنند، هر چند خریدار بليط با سیستم کار نمی‌کند و کاربر سیستم به شمار نمی‌آید ولی ذینفع محسوب می‌گردد و ضروری است انتظارات و خواسته‌های وی مدون گردد.

نمونه ۲: گرچه مدیر شبکه و پشتیبانی شرکت کارفرما، استفاده‌ای از سیستم حسابداری خواهد کرد، اما از آنجا که استقرار سیستم در شبکه و ساختار سخت‌افزاری تحت مدیریت وی انجام خواهد شد، ضروری است انتظارات وی در توسعه و ساخت سیستم مورد توجه قرار گیرد. به عنوان مثال مدیر شبکه و پشتیبانی اعلام می‌نماید که امکان افزایش سرویس‌دهنده‌ها وجود ندارد و سیستم نهایی باید قابل نصب در شرایط فعلی و بر روی سرویس‌دهنده‌های موجود باشد.

نمونه ۳: انتخاب مدیر عامل برای دوره بعدی در شهریورماه انجام خواهد شد. استقرار موفق سیستم خودکارسازی اداری^۲ امتیاز مهمی برای وی محسوب می‌شود و شانس وی برای انتخاب مجدد را افزایش خواهد داد. مدیر عامل انتظار دارد که سیستم خودکارسازی اداری قبل از شروع مردادماه عملیاتی شده باشد.

نمونه ۴: نمایندگان فروش بسته‌های نرم‌افزاری شرکت در بازار رضا و مجتمع کامپیوترا پایتحث در پروژه «بسته نرم‌افزاری حسابداری» ذینفع محسوب می‌گردد، چرا که توجه به انتظارات آن‌ها (به عنوان مثال نحوه بسته‌بندی و ظاهر جعبه حاوی نرم‌افزار) یکی از عوامل موقفيت پروژه خواهد بود.

1. Stakeholder
2. Office automation
3. Package

در واقع موقیت پژوهه وابسته به استخراج نیازهای ذینفعان و رفع آن‌هاست. بدیهی است برای موقیت پژوهه مشارکت فعالانه ذینفعان در فرآیند توسعه سیستم ضروری است. به عنوان مثال:

﴿ کاربران

کاربران باید دانش کسب و کار خود را در اختیار تیم توسعه قرار دهند و برای تعیین محدوده سیستم و اولویت‌بندی نیازمندی‌ها تصمیمات درست و به موقع اتخاذ کنند.

﴿ مدیران

ضروری است مدیران، تیم توسعه را حمایت و پشتیبانی نمایند. لازم است آن‌ها در ابتدا با فناوری و تکنیک‌های تیم توسعه آشنا شوند و به علت و تأثیر استفاده از آن‌ها توجه نمایند. این دانش به آن‌ها کمک خواهد کرد تا به شیوه‌ای درست و مؤثر در پیشبرد پژوهه مشارکت نمایند. مدیران نمی‌توانند تنها با مطالعه گزارش‌های هفتگی یا شرکت در جلسات ماهانه راهبری پژوهه، اطلاعات و دانش مذکور را کسب کنند. مدیران باید برای به دست آوردن اطلاعات و داده‌های پژوهه زمان کافی اختصاص دهند و فعالانه در فرآیند توسعه سیستم مشارکت داشته باشند.

﴿ تیم‌های استقرار و پشتیبانی

این تیم‌ها باید زمان، منابع و امکانات لازم را برای آشنایی اعضای خود با فناوری‌های مورد استفاده در ساخت سیستم اختصاص دهند. تیم پشتیبانی باید جزئیات سیستم و دلایل استفاده کاربران را به خوبی بشناسد، در غیر این صورت استقرار، راهاندازی، آموزش و استفاده کاربران از سیستم به شکست خواهد انجامید. ممکن است این کار با آموزش تیم پشتیبانی توسط تیم توسعه انجام شود. از طرف دیگر تیم استقرار باید به خوبی با نحوه نصب و راهاندازی سیستم آشنا باشد. ممکن است یک یا چند نفر از تیم استقرار در تیم توسعه شروع به کار کنند و در ادامه به تیم استقرار متقل شوند. فارغ از روش مشارکت، مشارکت مؤثر و برنامه‌ریزی شده تیم‌های استقرار و پشتیبانی با تیم توسعه الزامی است.

﴿ تیم توسعه سایر پژوهه‌ها

در صورت نیاز به یکپارچگی سیستم با سایر سیستم‌ها، اعضای تیم‌های آن‌ها باید همکاری لازم را داشته باشند. برای مثال در سیستم حساب سپرده کوتاه مدت برای دریافت اطلاعات مشتریان، مشارکت و همکاری تیم توسعه سیستم «اطلاعات مشتریان» ضروری است.

﴿ تیم نگهداری

ضروری است تیم نگهداری برای یادگیری سیستم با تیم توسعه کار کنند. هر چند ممکن است اعضاًی از تیم توسعه به تیم نگهداری متقل شوند اما لازم است دانش تیم توسعه به تیم نگهداری متقل شود.

-۸ مدیریت نیازمندی‌ها

یکی از معیارهای موفقیت پروژه‌های نرم‌افزاری، میزان انطباق عملکرد سیستم با نیازهای ذینفعان است. برای رفع نیازها راه حل‌هایی ارائه می‌شود که به شکل مجموعه‌ای از نیازمندی‌ها توصیف می‌شوند. برای بررسی میزان تطابق عملکرد سیستم با نیازهای ذینفعان (معیار موفقیت پروژه) ثبت و سازماندهی نیازمندی‌ها و توجه به تغییرات آن‌ها امری حیاتی است که آن را «مدیریت نیازمندی‌ها» می‌نامند. به عبارت دیگر «مدیریت نیازمندی‌ها فرآیند شناسایی، استخراج، مستندسازی، تحلیل، ردگیری، اولویت-بندي نیازمندی‌ها و کسب توافق بر سر آن‌ها و سپس کنترل تغییرات و تعامل با ذینفعان مرتبط با آن‌هاست». این فرایند، فرایند مستمر در طول پروژه است.

اندازه پروژه و پیچیدگی آن عوامل مهمی در تعیین میزان رسمیت مدیریت نیازمندی‌ها هستند. به عنوان مثال رسمی کردن «مدیریت نیازمندی‌ها در پروژه» دو نفره‌ای که تنها شامل ۱۰ نیازمندی است، غیرضروری است. اما در پروژه تولید نرم‌افزاری کوچک که شامل ۱۰۰ نیازمندی یا در پروژه تولید سیستم جامعی که شامل ۳۰۰۰ نیازمندی است، عدم سازماندهی، اولویت‌بندي، کنترل دسترسی و اختصاص منابع به نیازمندی‌ها، پروژه را با چالش‌های جدی مواجه خواهد کرد.

در اغلب پروژه‌ها معمولاً پرسش‌هایی شبیه موارد زیر مطرح می‌شوند:

- تغییر یک نیازمندی چه تاثیری بر سایر نیازمندی‌ها دارد؟
- چگونه می‌توان از پیاده‌سازی یک نیازمندی خاص در سیستم اطمینان حاصل کرد؟
- کدام یک از نیازمندی‌ها به تأیید ذینفعان رسیده است؟
- کدام ویژگی‌ها در نسخه اول نرم‌افزار پیاده‌سازی خواهند شد؟

برای پاسخ به پرسش‌های قبلی، راهی جز به کارگیری مفاهیم، تکنیک‌ها و استانداردهای مدیریت نیازمندی‌ها وجود ندارد. باید توجه داشت که مدیریت نیازمندی‌ها، ابداع جدیدی نیست، بلکه موضوعی است که هر تیم توسعه نرم‌افزار از گذشته تاکنون با آن مواجه بوده است و به روشهای نیازها و مشکلات خود را مرتفع نموده است.

هدف مدیریت نیازمندی‌ها، اطمینان از پوشش و صحت نیازها و انتظارات ذینفعان است. مدیریت نیازمندی‌ها با تحلیل و استخراج اهداف و قیدها شروع می‌شود و با برنامه‌ریزی، یکپارچه‌سازی و سازماندهی نیازمندی‌ها ادامه می‌یابد.

همچنین مدیریت نیازمندی‌ها شامل روش تعامل بین تیم پروژه و ذینفعان است. دلیل تأکید بر تعامل طرفین و چگونگی انجام آن، لزوم مشارکت کاربران و ذینفعان و ایجاد تفاهم بین طرفین در مورد تغییرات نیازمندی‌ها و کنترل آن در پروژه است.

به طور کلی مزایای اجرای مدیریت نیازمندی‌ها عبارتند از:

«کنترل مؤثر پروژه‌های پیچیده

یکی از مهمترین عوامل از کنترل خارج شدن پروژه‌ها، برداشت نادرست از رفتارهای سیستم در ابتدای پروژه و در ادامه تغییر تدریجی اما مداوم نیازمندی‌ها است. با مدیریت نیازمندی‌ها می‌توان از این موارد اجتناب کرد.

﴿ افزایش سطح کیفی و رضایتمندی مشتریان ﴾

یکی از اصلی‌ترین معیارهای کیفی سیستم، انجام درست کاری است که از آن انتظار می‌رود. این مهم تنها زمانی قابل سنجش و اندازه‌گیری است که کلیه ذینفعان در خصوصیات و عملکرد محصولی که ساخته خواهد شد، اتفاق نظر و تفاهم داشته باشند. مدیریت نیازمندی‌ها در ایجاد تفاهم نقش مؤثری ایفا می‌کند.

﴿ کاهش هزینه و تأخیر پروژه‌ها ﴾

همان‌طور که در فصل اول بیان شد، رفع خطاهای منتقل شده از مرحله نیازمندی‌ها به سایر مراحل، پرهزینه است. هدف فرآیند مدیریت نیازمندی‌ها، کاهش تعداد این گونه خطاهای است که منجر به کاهش هزینه و تأخیر زمانی پروژه می‌گردد.

﴿ افزایش و ارتقای تعامل اعضای پروژه ﴾

مدیریت نیازمندی‌ها با هدف اطمینان از درستی سیستم و پوشش کامل نیازهای ذینفعان، به دنبال مشارکت بهموقع ذینفعان در توسعه سیستم است. مشارکت بهموقع ذینفعان و پیروی از روش‌های پیشنهادی مدیریت نیازمندی‌ها، علاوه بر فراهم‌سازی زمینه شناخت درست و کامل سیستم، باعث ایجاد تفاهم بین تیم توسعه، کاربران، مشتریان و مدیران سیستم می‌شود.

۹- نکات کلیدی

- نیازمندی نرم‌افزاری شرط یا قابلیتی است که سیستم باید آن را پوشش دهد.
- یکی از روش‌های دسته‌بندی نیازمندی‌ها است.
- FURPS+
- انواع نیازمندی‌ها عبارتند از نیازهای ذینفعان، ویژگی‌های سیستم و نیازمندی‌های نرم‌افزاری.
- مدیریت نیازمندی‌ها فرآیند شناسایی، استخراج، مستندسازی، تحلیل، ردگیری، اولویت‌بندی نیازمندی‌ها و کسب توافق بر سر آن‌ها و سپس کنترل تغییرات و تعامل با ذینفعان مرتبط با آن‌هاست.

١٠ - مراجع

1. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
2. Leffingwell, Dean ,Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
3. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
4. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>



بخش دوم: تحلیل مسأله

در این بخش چگونگی تحلیل مسأله در یک پروژه توسعه نرم افزار تشریح می‌گردد. مسأله، عاملی است که برای حل آن، نرم افزاری طراحی و پیاده سازی می‌گردد. هدف از تحلیل مسأله، کسب توافق بر سر مسأله‌ای است که قرار است حل شود و شامل شناسایی ذینفعان، تعریف مرز سیستم و شناسایی قیدهای حاکم بر سیستم است.

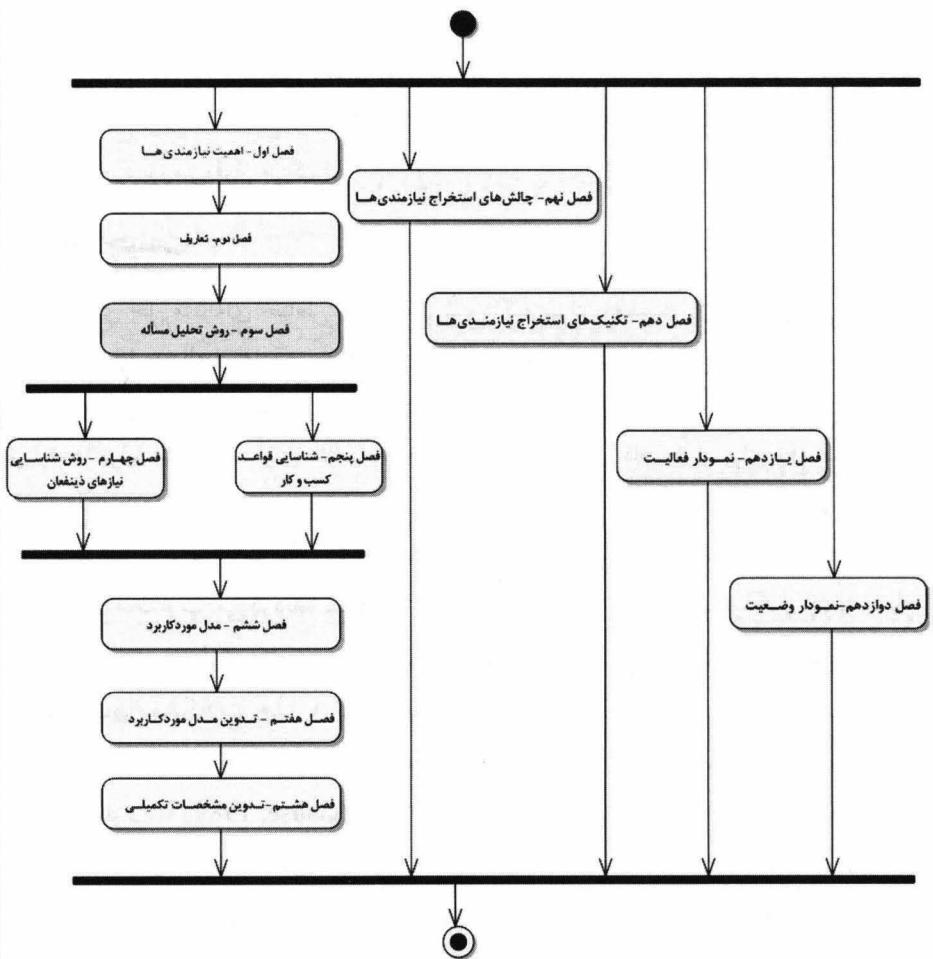
فصل سوم – روش تحلیل مسأله

فصل سوم - روش تحلیل مسئله

پاسخ به پرسشی که فهمیده نشده باشد، کاری ابلاوهانه است. کار کردن برای رسیدن به هدفی که مورد پسند نیست، غم انگیز است.

جورج پولیا

«همچو کتابی است جهان، جامع احکام نهان
جان تو سردفتر آن، فهم کن این مسئله را»
مولوی



۱- مقدمه

هدف این فصل، بررسی چگونگی شناسایی و استخراج نیازهای واقعی ذینفعان سیستم، توسط تیم توسعه است. سیستم نرم‌افزاری برای پاسخ‌گویی و ارائه راه حل به مسائل ذینفعان توسعه داده می‌شود. شناسایی مسأله، توافق آن با ذینفعان، تعیین مرز سیستم و شناسایی قیدهای حاکم بر راه حل، نقش اساسی در پاسخ‌گویی به مسائل ذینفعان ایفا می‌کند. در این فصل روش تحلیل مسأله و موضوعات مرتبط با آن مورد بررسی قرار می‌گیرد.

۲- مسأله چیست

هر سیستم برای حل مسأله‌ای خاص ایجاد می‌شود که برای اطمینان از شناسایی درست آن از تکنیک‌های تحلیل مسأله استفاده می‌شود. مسأله تنها به مفهوم وجود مشکل نیست و لزوماً تمامی برنامه‌ها برای حل مشکل ایجاد نمی‌شوند، بلکه برخی از آن‌ها برای استفاده از فرصت‌های تجاری یا تحقق ایده‌ها تولید می‌گردند. برای مثال، برنامه‌هایی مانند بازی‌های کامپیوتری برای رفع مشکل تولید نمی‌شوند بلکه با هدف کسب درآمد توسعه می‌یابند.

فرصت‌ها و مشکلات دو روی یک سکه هستند: مشکلات مشتریان فرصت‌های تولیدکنندگان محسوب می‌گردند. بسته به این که از کدام طرف به سکه نگاه کنید، نام آن عوض می‌شود. از آنجا که اکثر سیستم‌ها مشکلات قابل شناسایی را پوشش می‌دهند می‌توان از جداول فرصت‌ها/مشکلات حذر و تنها بر روی مشکلات تمرکز کرد.

«مسأله عبارت است از تفاوت بین آنچه هست (وضعیت فعلی) با آنچه انتظار می‌رود (وضعیت مطلوب)» (Gause, 1989).

توسعه‌دهنگان همواره گمان می‌کنند «مشکل واقعی» این است که «کاربران نمی‌دانند مشکل واقعی چیست». این موضوع یکی از مضلاع مشترک توسعه سیستم‌های نرم‌افزاری است. تعریف قبل این معضل را رفع می‌کند یا تخفیف می‌دهد. بر اساس این تعریف، موضوعی که از طرف کاربر به عنوان مشکل اعلام می‌گردد باید توسط تیم توسعه به عنوان «مسأله واقعی» پذیرفته شود. توجه نمایید که این به معنای قبول و پذیرش هر موضوع اعلام شده توسط کاربر نیست، بلکه تأکیدی بر به رسمیت شناختن جایگاه کاربر به عنوان کسی است که تفاوت وضعیت فعلی و وضعیت مطلوب خود را درک می‌کند. گاه کاربران به دلایلی

از جمله ناآشنایی با امکانات نرمافزاری، تصور درستی از وضعیت مطلوب ندارند و تنها قادر به بیان وضعیت فعلی هستند^۱.

از تعریف قبلی می‌توان نتیجه گرفت که تغییر فرایند فعلی بدون تولید سیستم نرمافزاری نیز می‌تواند راه حل قابل قبولی باشد.

«فرایند شناسایی مشکلات واقعی و نیازهای کاربران و ارائه راه حل برای رفع آنها را تحلیل مسأله می‌نامند».

در تحلیل مسأله، حوزه مسأله شناخته می‌شود و تحلیل و راه حل قابل انجام ارائه می‌گردد. عموماً چندین راه حل برای مسأله پیشنهاد می‌گردد که وظیفه تیم توسعه انتخاب یکی از آنها است. کسب توافق در مورد راه حل انتخاب شده یکی از چالش‌های پروژه است. افزایش میزان آگاهی کاربر از واقعیت‌ها، خواسته‌ها و انتظاراتش، مقرنون به صرفه‌ترین رویکرد در رفع چالش مذکور است. تجربه نشان داده است که افزایش آگاهی کاربران منجر به انتخاب راه حلی ارزان، سریع و باکیفیت می‌گردد. راه حل انتخاب شده لزوماً تولید سیستم جدید نیست، بلکه می‌تواند بهبود تدریجی سیستم ناقص فعلی یا آموزش مجدد کاربران در استفاده از آن باشد.

در شرایطی که شکاف بین واقعیت‌ها و انتظارات کاربر را نتوان با روش‌هایی غیر از توسعه سیستم جدید کاهش داد، تعریف و طراحی سیستم جدید ناگزیر است. سیستم جدید باید بتواند وضعیت موجود را به وضعیت مطلوب تبدیل کند یا فاصله بین دو وضعیت را حداقل نماید. در نتیجه چالش بزرگ در توسعه سیستم عبارت خواهد بود از «چگونه فاصله و مغایرت بین واقعیت‌های موجود و انتظارات کاربر توسط سیستم کاهش یابد؟». در این حالت هدف از تحلیل مسأله شناخت صحیح مشکل، قبل از آغاز فرایند توسعه است که شامل گام‌های زیر است:

- ۱- توافق بر سر تعریف مسأله
- ۲- شناسایی عوامل بروز مسأله
- ۳- شناسایی ذینفعان و کاربران
- ۴- تعیین مرز سیستم
- ۵- شناسایی قیدهای راحل

در ادامه فصل هر یک از این مراحل بررسی می‌شوند.

۱- بی اطلاعی کاربر از کاربرد نرمافزار در بهبود وضعیت فعلی منجر به ناتوانی وی در چگونگی ارائه وضعیت فعلی می‌گردد. ناگفته پیداست که تشریح وضعیت فعلی نیز به طور کامل میسر نیست.

۲- کاربرد تحلیل مسأله تنها در توسعه سیستم‌های جدید نیست، در اینجا تاکید بر استفاده از آن در توسعه سیستم‌های جدید است.

۳- توافق بر سر تعریف مسئله

اولین گام در تحلیل مسئله، توافق بر سر تعریف مسئله‌ای است که انتظار حل آن می‌رود. یکی از ساده‌ترین راه‌ها برای دستیابی به توافق طرفین این است که: «مسئله را مطرح کنید. آیا دیگران با آن موافق هستند؟».

شناخت و تدوین مزایای راه حل در پیشبرد فرایند حل مسئله بسیار مفید است. از این رو توصیه می‌گردد علاوه بر تعریف مسئله مزایای راه حل نیز مدون گردد. مزایای راه حل باید به زبان و با ادبیات کاربران و ذینفعان سیستم نوشته شود. این کار به شناسایی ابعاد بیشتری از مسئله کمک می‌کند. جدول زیر استانداردی برای تدوین تعریف مسئله و مزایای راه حل است. تدوین صورت مسئله بدین شکل به کسب اطمینان از «هدف مشترک» تمامی ذینفعان کمک می‌کند.

جدول ۱-۳: قالب تعریف مسئله

مسئله	[شرح و توضیح مسئله]
که تأثیر می‌گذارد بر	[فهرست ذینفعانی که تحت تأثیر مسئله هستند]
و تأثیرات آن عبارتند از	[شرح تأثیرات مسئله بر روی ذینفعان و کسب و کار آنها]
اجرای راه حل موفق خواهد توانست	[لیست مزایای کلیدی راه حل]

جدول بالا عبارت زیر است که به شکل جدول مدون شده است:
[مسئله [...] که تأثیر می‌گذارد بر [...] و تأثیرات آن عبارتند از [...]. اجرای راه حل موفق خواهد توانست [...]].

این جدول در سنده «چشم‌انداز^۱» تدوین می‌گردد.

مثال زیر از سیستم «حساب سپرده کوتاه مدت» انتخاب شده است.

جدول ۲-۳: نمونه تعریف مسئله

مسئله	رویه دستی امور شعبه‌ای در شعب بانک
که تأثیر می‌گذارد بر	مشتریان بانک، رئیس و پرسنل شعبه، پرسنل ستاد
و تأثیرات آن عبارتند از	نارضایتی مشتریان از عدم امکان دریافت خدمات از سایر شعب و دشواری نظارت بر شعب بانک
اجرای راه حل موفق خواهد توانست	به همراه افزایش رضایتمندی مشتریان، تعداد مشتریان را افزایش دهد و موجب افزایش مزیت‌های رقابتی با سایر بانک‌ها، کنترل و نظارت متتمرکز و دقیق‌تر بر شعب، تسريع در ارائه داده‌ها و گزارش‌های تحلیلی به هیات مدیره و مدیریت ارشد گردد.

۱- این سنده و اجزای آن در فصل‌های بعدی معرفی می‌گردد.

۴- شناسایی عوامل بروز مسأله

در این بخش به بررسی عوامل بروز مشکل یا نیاز مشتری، اهمیت شناسایی، روش‌های شناسایی و تدوین آن‌ها پرداخته می‌شود.

۴-۱- اهمیت شناسایی عوامل بروز مسأله

می‌گویند روزی بیماری به پزشک مراجعه کرد که از درد تمام اعضای بدنش می‌نالید. دکتر از بیمار پرسید «کجا بدن درد می‌کند؟». بیمار با انگشت اشاره، تک‌تک اعضای بدنش را لمس کرده و با ناله‌هایش به پزشک عارض شد که «آقای دکتر همه جای بدن درد می‌کند. اینجا آخ... اینجا آخ...». دکتر با معاینه‌ها و آزمایش‌های فراوان علت درد را نیافت و آن را مرضی لاعلاج تشخیص داد. تا این که به طور اتفاقی پی‌برد که مشکل در انگشت بیمار است. با معاینه‌ای مختصراً درد را ناشی از خار کوچکی یافت که در انگشت او فرو رفته بود. ناله‌های بیمار نه به خاطر درد اعضای بدنش که به واسطه درد انگشت اشاره‌اش بود.

در داستان مذکور «مسأله» درد تمام اعضای بدن بیمار و «عامل بروز مسأله» خار نوک انگشت او بود. در توسعه سیستم‌های نرم‌افزاری واقعی مشابه داستان مذکور بسیار رخ می‌دهد. مشتری در نقش یک بیمار به تیم توسعه مراجعه و از مسائلی یاد می‌کند که وی را به سته آورده است و برای رفع آن‌ها تقاضای توسعه سیستم دارد. تیم توسعه مانند پزشک باید خار سر انگشت را شناسایی و درمان نماید. مسأله‌های مشتریان باعث بروز نیازهای آنان می‌گردد. با این دیدگاه می‌توان نیازهای مشتریان را به دو دسته تقسیم نمود: نیازهای بیان شده^۱ و نیازهای واقعی^۲. نیازهای بیان شده درد همه اعضای آن بیمار و نیازهای واقعی درد انگشت اشاره او است. لازم است سیستم نرم‌افزاری برای حل مشکلات کاربران به جای نیازهای بیان شده، نیازهای واقعی آنان را رفع نماید. در تحلیل مسأله برای شناسایی نیازهای واقعی باید به دنبال علت اصلی^۳ مشکلات و نیازهای ذینفعان بود و به اظهارات آن‌ها اکتفا نکرد. بی‌توجهی به این موضوع، موجب تولید سیستمی ناکارآمد خواهد شد که بی‌شک به دلیل عدم رفع نیازهای واقعی موجب دویاره‌کاری و حتی عدم استفاده از نرم‌افزار می‌شود.

بسیاری از اوقات «شناسایی عوامل اصلی بروز مسأله» به «شناسایی مسأله پشت مسأله»^۴ نیز تعبیر می‌گردد، چرا که تیم توسعه باید شرح «مشکلات و نیازها» را از ذینفعان و کاربران استخراج کند، ولی به دنبال «دلیل اصلی بروز مشکل یا نیاز» باشد.

1. Stated needs

2. Real needs

3. Root cause

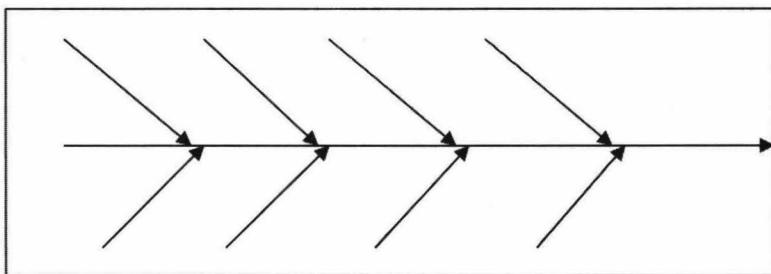
4. Problem of problem

۴-۲- روش شناسایی عوامل اصلی

پس از شناسایی درست مسئله از تکنیک‌های مختلفی برای شناسایی عوامل آن می‌توان استفاده نمود. این تکنیک‌ها، روش‌های سیستمی برای شناسایی دلایل اصلی وجود مسئله هستند. «نمودار استخوان ماهی» و «اصل پارتو» از جمله این روش‌ها هستند که در ادامه مورد بحث و بررسی قرار می‌گیرند.

۴-۳- نمودار استخوان ماهی

نمودار استخوان ماهی توسط دکتر کالورو ایشیکاوا^۱ از پیشگامان مدیریت کیفیت در دهه ۶۰ میلادی ابداع شده است. این نمودار را «نمودار ایشیکاوا» نیز می‌نامند که به دلیل شباهت به استخوان‌های ماهی به این نام نیز شهرت دارد. این نمودار ابزاری تحلیلی است که به شیوه‌ای سیستمی عوامل تأثیرگذار بر یک موضوع را بررسی می‌کند. از این رو آن را نمودار علت-معلول^۲ نیز می‌نامند.



شکل ۳-۱: نمودار استخوان ماهی

هدف از کاربرد نمودار استخوان ماهی یافتن راه حلی برای مسئله نیست، بلکه هدف شناسایی علل اصلی بروز آن است. به عبارت دیگر در این روش تأکید بر عوامل بروز مسئله به جای خود آن است. در این شیوه از ابزارهای گرافیکی برای برقراری ارتباط بین مسئله و عوامل ایجاد آن استفاده می‌گردد. در شرایط زیر می‌توان از این روش استفاده نمود:

- برای تحلیل مسائل پیچیده
- زمانی که چندین عامل برای بروز مسئله وجود دارد
- زمانی که روش‌های سنتی حل مسئله مانند روش سعی و خطابسیار زمانبر یا هزینه‌بر باشند بدیهی است این شیوه در همه موقعیت‌ها قابل استفاده نیست. توصیه می‌گردد که از این روش در موارد زیر استفاده نشود:
- وقتی که مسئله ساده یا شناخته شده باشد
- تعداد و ترکیب افراد برای شناسایی عوامل کافی نباشد

1. Kaoru Ishikawa
2. Cause-and-Effect

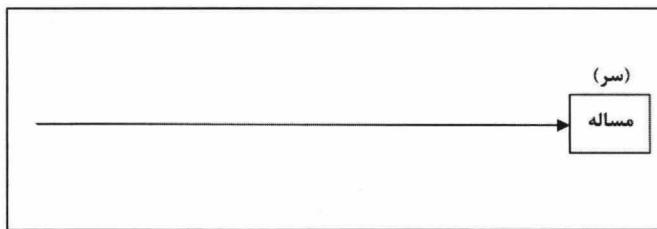
- مسئله به سادگی توسط افراد خبره قابل حل باشد

۴-۱-۳-۴- ترسیم نمودار استخوان ماهی

گام‌های ترسیم نمودار استخوان ماهی عبارتند از:

- شناسایی و توافق بر سر مسئله
- شناسایی گروه‌های عوامل بروز مسئله
- شناسایی عوامل بروز مسئله

در گام اول ابتدا مسئله مشخص می‌گردد. بعد از آن مسئلول جلسه ترسیم نمودار را شروع می‌کند. برای این کار مسئله در سمت راست صفحه درون یک مستطیل نوشته می‌شود. سپس از سمت چپ صفحه، پیکانی مستقیم تا مستطیل ترسیم می‌گردد. در انتهای این گام، تصویری مانند شکل زیر به دست می‌آید.



شکل ۲-۳: تصویر پایان گام اول

هدف این گام شناسایی گروه‌های عوامل بروز مسئله است. برای این کار می‌توان از روش توفان ذهنی^۱ استفاده نمود^۲. در صورتی که گروه‌بندی دشوار باشد یا نتیجه مناسب از این روش به دست نیاید می‌توان از گروه‌بندی‌های عمومی استفاده کرد. یکی از گروه‌بندی‌های عمومی در ادامه معرفی شده است:

- روش^۳: عواملی در این گروه قرار می‌گیرند که مرتبط با فرایند انجام کار باشند.
- افراد^۴: عوامل انسانی تأثیرگذار در این گروه قرار می‌گیرند.
- مدیریت^۵: این گروه شامل عواملی است که به موضوعات و تصمیم‌گیری‌های مدیریتی مرتبط هستند.

1. Brainstorming

2- این روش در فصل‌های آینده به تفصیل معرفی شده است.

3. Method

4. Man

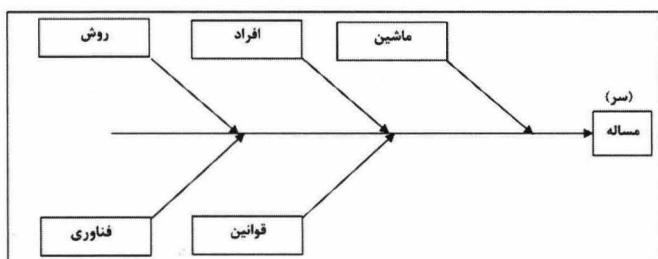
5. Management

- سنجش^۱: عوامل این گروه مربوط به اندازه‌گیری‌ها و روش‌های آن‌ها هستند. ممکن است عوامل مذکور ناشی از سنجش‌های نامناسب یا نادرست یا روش‌های مورد استفاده باشد.
 - ماده^۲: عوامل مرتبط با مواد مورد استفاده در این گروه جای داده می‌شوند.
 - ماشین^۳: عوامل این گروه به ابزارهای مورد استفاده اشاره دارند.
- گروه‌های معرفی شده تنها برای نمونه ارائه شده‌اند. کاربرد گروه‌های مذکور، تحلیل عوامل مسأله است که لزوماً در محدوده توسعه نرم‌افزار نخواهد بود. به عنوان مثال چنانچه مسأله سازمان، بهره‌وری نامناسب باشد و عوامل آن ابزارهای نامناسب و انگیزه پایین کارکنان باشد، رفع عوامل مذکور در محدوده توسعه سیستم‌های نرم‌افزاری نیست.
- استفاده از گروه‌های عمومی در زمان شناسایی عوامل این خطر را به همراه دارد که باعث جهت‌دهی فکری شرکت‌کنندگان گردد.

شناسایی عوامل بروز مسأله

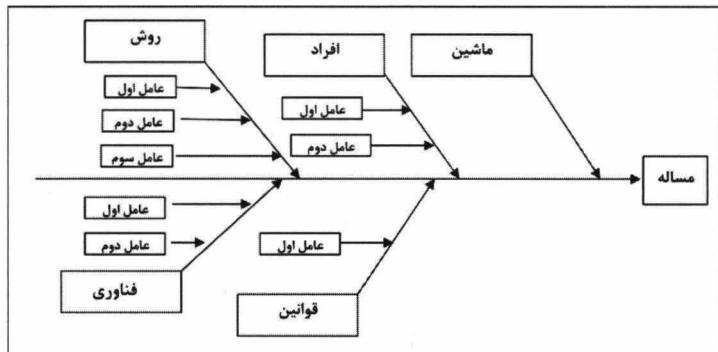
در ابتدای جلسه توفان ذهنی، گروه‌های عوامل بروز مسأله مورد بررسی قرار می‌گیرند. ابتدا یک گروه انتخاب و شناسایی عوامل با بحث و بررسی آن شروع می‌شود. این کار برای تمامی گروه‌ها تکرار می‌گردد. این شیوه موجب تمکرک تیم تحلیل مسأله بر دامنه محدودی از مسأله‌ها می‌شود که منجر به تصفیه و پالایش عوامل مرتبط می‌گردد.

توصیه می‌گردد که حداقل چهار تا شش گروه از گروه‌های عوامل بروز مسأله انتخاب گردند. پس از انتخاب، گروه‌های انتخاب شده مانند استخوان‌های ماهی به بدنه نمودار متصل می‌گردند. برای اتصال از پیکان‌های مورب استفاده می‌گردد که سر آن‌ها به سمت ستون فقرات ماهی است. شکل نهایی مانند شکل زیر خواهد بود.



شکل ۳-۳: تصویر پایان گام دوم

در ادامه، عوامل شناسایی شده از هر گروه، روی پیکانی نوشته می شود که به پیکان گروه مربوطه اش متصل است. تصویر نهایی مانند شکل زیر خواهد بود.



شکل ۳-۴: اتصال عامل‌ها به دسته متناظر

پس از اتمام این مرحله مجموعه‌ای از عوامل در هر گروه وجود خواهد داشت. شناسایی عوامل می‌تواند با تکرار این مرحله روی هر یک از عوامل ادامه یابد که منجر به شناسایی عوامل جزئی تری خواهد گردید. زمانی که اطمینان حاصل شود عوامل اصلی مسأله شناسایی شده‌اند، کار متوقف می‌گردد.

۴-۳-۲- اصل پارتو

پس از شناسایی عوامل، رفع تمامی آن‌ها ایده‌آل است، اما تجربه نشان می‌دهد که هزینه رفع تمامی عوامل بیش از هزینه‌ای است که خود مسأله به ذینفع تحمیل می‌کند. از این رو رفع تمامی عوامل بروز مسأله مقرنون به صرفه نیست، ناگزیر باید تعدادی از آن‌ها انتخاب گردد. اهمیت عوامل و سهم آن‌ها در بروز مسأله، معیارهای مناسبی برای انتخاب است. برای این کار از اصل پارتو استفاده می‌گردد. در سال ۱۹۰۶ اقتصاددان ایتالیایی ویلفردو پارتلو^۱ در حین تحقیق راجع به توزیع ثروت در ایتالیا پی‌برد که ۸۰ درصد ثروت کشورش متعلق به ۲۰ درصد از مردم است. او ۲۰ درصد ثروتمند را «اقلیت مهم» و ۸۰ درصد دیگر را «اکثربت کم اهمیت» نامید.

دکتر ژوزف جوران^۲ در سال ۱۹۴۱ به صورت اتفاقی کارهای پارتو را بررسی کرد و در مباحث کیفیت به کار گرفت. وی کارهای پارتو را به صورت اصلی ارائه نمود و آن را اصل پارتو یا اصل ۲۰-۸۰ نامید.^۳

1. Vilfredo Pareto

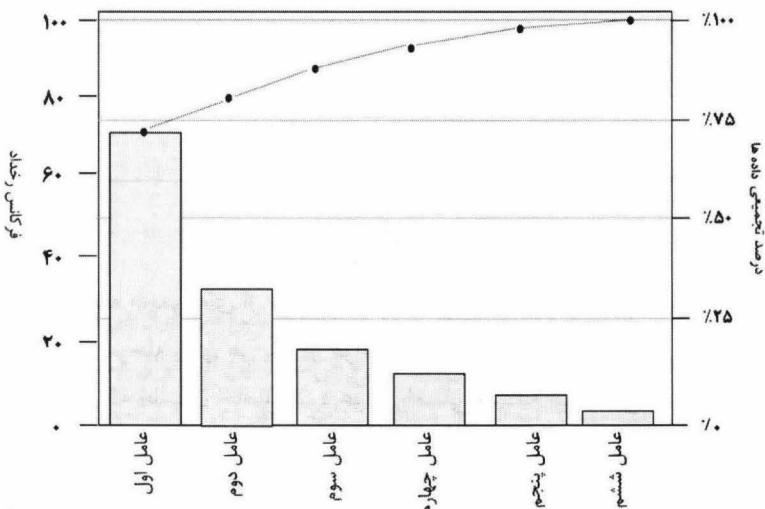
2. Joseph M. Juran

3- سال‌ها بعد جوران اصل را به شکل «اقلیت مهم و اکثربت مفید» تغییر داد تا نشان دهد که ۸۰ درصد عوامل باقی مانده باید به طور کلی نادیده گرفته شوند.

از اصل پارتو می‌توان نتیجه گرفت که ۸۰ درصد مسأله، ناشی از ۲۰ درصد عوامل است. از این‌رو تنها ۲۰ درصد عوامل شناسایی شده با روش استخوان ماهی حائز اهمیت خواهد بود. عوامل مهم را به کمک نمودار پارتو^۱ می‌توان شناسایی نمود. نمودار پارتو دارای گرافی میله‌ای^۲ و گرافی خطی^۳ است.

داده‌های عوامل به شکل نزولی و به صورت گراف میله‌ای و درصد تجمعی داده‌ها به صورت گراف خطی ترسیم می‌گردد.

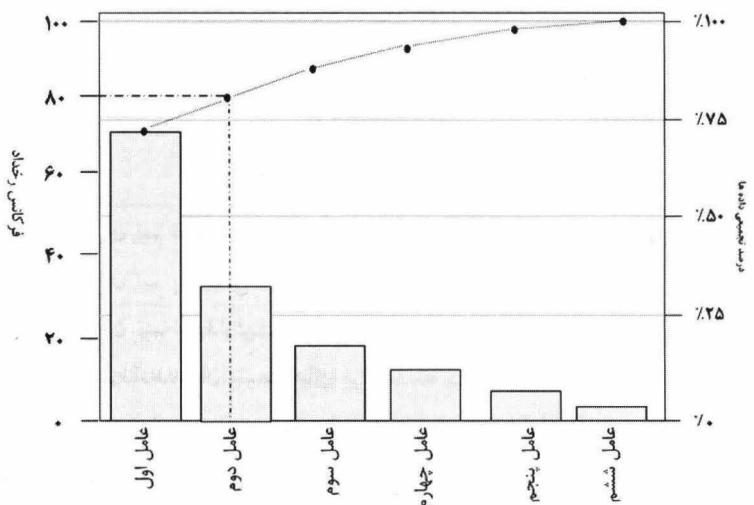
داده‌ها می‌توانند فرکانس رخداد، هزینه‌ها یا هر موضوع قابل اندازه‌گیری مرتبط با عوامل باشند.



شکل ۳-۵: نمودار پارتو

مطابق شکل ۳-۶، برای شناسایی «عوامل مهم» کافی است نقطه‌ای در گراف درصد تجمعی مشخص گردد که مقدار آن در محور سمت راست برابر با ۸۰٪ باشد. از آن نقطه خطی عمود به محور عوامل ترسیم می‌گردد. تمامی عواملی که در سمت چپ قرار گیرند «عوامل مهم» محسوب خواهند گردید چرا که علت بروز ۸۰ درصد از مسأله هستند.

1. Pareto Chart
2. Bar graph
3. Bar graph



شکل ۳-۶: شناسایی عوامل اصلی در نمودار پارتو

۵- شناسایی ذینفعان و کاربران

در این بخش روش شناسایی ذینفعان و کاربران سیستم و اهمیت شناسایی آن‌ها تشریح می‌گردد.

۱-۵- شناسایی ذینفعان

در فصل دوم گفته شد ذینفع کسی است که تحت تأثیر پیاده‌سازی سیستم قرار خواهد گرفت. به عبارت دیگر، ذینفعان، گروهی هستند که لازم است نیازها و انتظارات آنان در مراحل توسعه سیستم مدنظر قرار گیرند و برآورده شوند. ذینفعان تنها شامل کاربران سیستم نیستند، بلکه گروههای را نیز شامل می‌شوند که هرچند به صورت مستقیم از سیستم استفاده نمی‌کنند، اما تحت تأثیر توسعه سیستم خواهند بود. این دسته از ذینفعان را باید در کسب‌وکار یا محیط‌های اجرایی و عملیاتی سیستم جستجو کرد.

لازم است که مراحل توسعه سیستم و محصول نهایی توسط ذینفعان مورد پذیرش قرار گیرد. موقفيت پروژه در گروه تایید آن توسط ذینفعان خواهد بود. چنانچه ذینفعان سیستم یا معیارهای پذیرش آن‌ها به درستی شناسایی نگرددند، امکان عدم پذیرش سیستم توسط آن‌ها وجود دارد. از این‌رو انتظار می‌رود تیم توسعه، ذینفعان و معیارهای پذیرش آنان را به درستی شناسایی و در طول توسعه سیستم دنبال نماید. معمولاً ذینفعان انتظارات متفاوت و گاه متناقضی نسبت به توسعه سیستم و محصول دارند که باید مدیریت شوند. یکی از چالش‌هایی که تیم توسعه با آن روپرست، رفع تناقض و همسو کردن انتظارات ذینفعان است.

هدف از این گام در حل مسأله، شناسایی ذینفعان و انتظارات آنان، تدوین معیارهای آنان در پذیرش

سیستم، رفع تناقض و همسو کردن انتظاراتشان است.

در پروژه‌های توسعه نرم‌افزار، گروه‌های زیر کاندیداهای مناسبی برای ذینفعان سیستم هستند:

- کاربران
 - مدیران
 - اعضا و مدیران تیم نصب و استقرار^۱
 - اعضا و مدیران تیم پشتیبانی
 - اعضا و مدیران تیم نگهداری
 - مدیران و توسعه‌دهندگان سیستم‌های مرتبط
- بنا بر میزان تسلط تیم توسعه بر حوزه مسئله، شناسایی ذینفعان می‌تواند کاری ساده و بدیهی یا کاری بسیار سخت و دشوار باشد. پاسخ‌گویی به پرسش‌های زیر می‌تواند به شناسایی ذینفعان کمک کند:
- کاربران سیستم چه کسانی هستند؟
 - مشتریان یا خریداران سیستم چه کسانی هستند؟
 - چه کسانی تحت تأثیر خروجی‌های سیستم قرار می‌گیرند؟
 - چه کسانی در زمان نصب، راه اندازی و تحويل، سیستم را بررسی و تأیید می‌کنند؟
 - چه کسانی سیستم جدید را نگهداری و پشتیبانی خواهند کرد؟
- لازم است مشخصات ذینفعان پس از شناسایی آنان مستند گردد. این کار در مستند «سندهای انداز»^۲ به شکل زیر انجام می‌شود:

جدول ۳-۳: جدول توصیف ذینفعان

نام	شرح	مسئولیت‌ها

شرح هر یک از ستون‌ها در زیر آمده است.

- نام: نام یا عنوان ذینفعان
- شرح: متن کوتاهی برای معرفی ذینفعان
- مسئولیت‌ها: خلاصه‌ای از مسئولیت‌های اصلی ذینفع در مقابل توسعه سیستم به عنوان مثال برای سیستم «حساب سپرده کوتاه مدت» این جدول به شکل زیر است.

۱- با این فرض که تیم‌های نصب و استقرار، پشتیبانی و نگهداری مستقل از تیم توسعه پروژه هستند.

2. Vision

۲- در سندهای انداز بخشی به نام پروفایل ذینفعان وجود دارد که حاوی اطلاعات مفصلی درباره آنان است. هم‌چنین سندهای انداز خواسته‌های ذینفعان (Stakeholder Requests) نیز می‌توانند در جمع‌بندی خواسته‌های ذینفعان مفید باشد.

جدول ۳-۴: جدول ذینفعان سیستم حساب سپرده کوتاه مدت

نام	شرح	مسئولیت‌ها
متصلی امور بانکی	-	ارائه نیازهای شعبه، تأیید نمونه اولیه رابط کاربر، آزمون و تأیید سیستم
سرپرستی امور شعب	-	ارائه نیازهای کنترلی و مدیریتی، تأیید نمونه اولیه رابط کاربر، آزمون و تأیید سیستم

توسعه‌دهندگان نیز ذینفع محسوب می‌شوند، اما در فهرست ذینفعان قرار نمی‌گیرند. زیرا همان‌طور که پیشتر بیان شد ذینفعان منشأ نیازهای سیستم هستند در حالی که توسعه‌دهندگان نمی‌توانند نیازی به نیازهای سیستم بیفزایند و اقدام به پیاده‌سازی آن کنند. توسعه‌دهندگان می‌توانند نیازهای مد نظر خود را به ذینفعان پیشنهاد کنند و پس از کسب توافق، آن‌ها را در زمرة نیازهای سیستم محسوب نمایند. ذینفع، بیانگر مجموعه‌ای از افراد حقیقی یا حقوقی است. به عنوان مثال بانک دارای چندین متصلی امور بانکی است که در جدول ذینفعان تنها سمت آن‌ها آورده می‌شود. ارتباط با تمامی افراد مشمول نقش یک ذینفع کاری دشوار و غیرضروری است، لذا تیم توسعه معمولاً با نماینده یا نمایندگان معدودی از ذینفعان در ارتباط خواهد بود. لازم است نمایندگان ذینفعان دارای ویژگی‌های زیر باشند:

۱. تسلط به حوزه کسب‌وکار یا حداقل بخشی از آن

نمایندگان ذینفعان باید بر حوزه مسأله یا بخشی از آن مسلط باشند. این افراد خبره حوزه مسأله^۱ یا خبره موضوع^۲ نامیده می‌شوند.

۲. تفکر منطقی

لازم است نمایندگان ذینفعان علاوه بر درک کسب‌وکار و آشنایی کامل با آن، به شکل سیستمی و در قالب یک روش منطقی فرایندها و کارهای حوزه مسأله را توصیف نمایند.

۳. مهارت‌های ارتباطی

نماینده ذینفعان باید خصوصیات رفتاری و مهارت‌های ارتباطی مناسبی برای انجام کار تیمی و همکاری با توسعه‌دهندگان داشته باشد.

۴. علاقه‌مند به همکاری

اغلب، کار اصلی ذینفعان کاری غیر از همکاری در توسعه نرم‌افزار است. توجیه آنان نسبت به اهمیت و اثرات مشارکتشان در توسعه سیستم از وظایف تیم توسعه است. سوابق بد همکاری در توسعه نرم‌افزار و ذهنیت منفی از همکاری‌های قبلی، یکی از

1. Domain expert

2. Subject matter expert

عوامل عدم همکاری مؤثر ذینفعان است. انجام به موقع تعهدات در رفع این معضل و جلب اعتماد ذینفعان مؤثر است.

۵. جامع‌نگری و دگراندیشی

اشخاصی که دارای مشخصه‌های زیر باشند، کاندیدای مناسبی برای نمایندگی ذینفعان خواهند بود:

- اشخاصی که تنها از یک زاویه به موضوع می‌نگرند و کار خود یا واحد ذیربسط را مهم‌تر از سایر کارها می‌دانند.
- اشخاصی که تنها قادر به اندیشیدن در چارچوب شیوه جاری هستند، علاقه یا توانایی تفکر به شیوه‌ای متفاوت را ندارند.
- اشخاصی که در تعیین اولویت نیازهایشان ناتوانند.

۶. شناخت جایگاه ذینفع در توسعه نرم‌افزار

لازم است نمایندگان ذینفعان از جایگاه، نقش و مسئولیت‌های خود در توسعه نرم‌افزار آگاه باشند و بدانند در صورت انجام نادرست مسئولیت‌هایشان موفقیت پژوهه را به خطر خواهند انداخت.

۲-۵- شناسایی کاربران و محیط کاربری

کاربران سیستم کسانی هستند که با سیستم کار می‌کنند و در دسته‌های خاص از ذینفعان می‌گنجند. شناسایی کاربران با هدف تعیین محدوده سیستم و ارائه راه حل پیشنهادی صورت می‌گیرد. اهمیت شناسایی کاربران در فصل ششم با جزئیات بیشتری بیان خواهد شد. کاربران نیز در «سند چشم‌انداز» مستندسازی می‌شوند:

جدول ۳-۵: جدول توصیف کاربران

نام	شرح	فعالیت‌ها	ذینفع

- ﴿ نام: نام یا عنوان کاربر
- ﴿ شرح: متن کوتاهی برای معرفی کاربر
- ﴿ فعالیت‌ها: کارهایی که کاربر با سیستم انجام خواهد داد. کارهای مذکور به تدریج در ادامه روند پژوهه تکمیل خواهند گردید.
- ﴿ ذینفع: نام ذینفع مرتبط با کاربر. این اطلاع، زمانی مفید است که ذینفع دارای بیش از یک کاربر در سیستم باشد. در صورتی که نام ذینفع و کاربر یکی باشد، می‌توان از تکرار آن خودداری کرد. به عنوان مثال به نمونه زیر توجه کنید:

جدول ۳-۶: جدول کاربران سیستم حساب سپرده کوتاه مدت

نام	شرح	فعالیت‌ها	ذینفع
متصدی امور بانکی	-	انجام عملیات شعبه‌ای شامل تعریف مشتری، افتتاح حساب، واریز، برداشت، مسدود کردن و رفع مسدودی حساب	-
کارشناس سرپرستی امور شعب	-	تعریف شعبه و تعریف پرسنل	سرپرستی امور شعب
مدیر سرپرستی امور شعب	-	گزارش عملکرد شعب	سرپرستی امور شعب

پس از شناسایی کاربران، ضروری است مشخصات محیط استفاده نرم‌افزار، مشخص و توصیف گردد. تدوین این موضوع در بخش «محیط کاربران» سند چشم‌انداز انجام می‌گیرد. پاسخ به پرسش‌های پیشنهادی زیر می‌تواند تکمیل این بخش را تسريع نماید.

- تعداد افرادی که کار را انجام می‌دهند؟ آیا این تعداد متغیر است؟
- انجام کل کارها چه مدت به طول می‌انجامد؟ مدت زمان هر گام چه اندازه است؟ آیا این زمان متغیر است؟
- مشخصات منحصر به فرد محیط کدامند؟(به عنوان مثال استفاده در هنگام پرواز یا حرکت در خودرو)
- فناوری‌هایی که در حال حاضر مورد استفاده قرار می‌گیرند، کدامند؟ آیا طرحی برای تغییر آن‌ها وجود دارد؟
- نرم‌افزارهای دیگری که در حال حاضر مورد استفاده قرار می‌گیرند، کدامند؟ آیا نیاز به یکپارچه‌سازی سیستم با آن‌ها وجود دارد؟

به عنوان مثال در سیستم حساب سپرده کوتاه مدت می‌توان چنین نوشت:

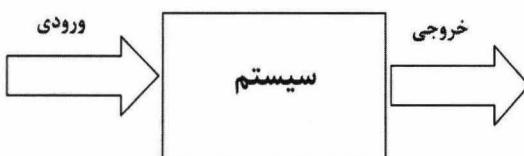
« محل استفاده سیستم تنها در شب بانک و اداره کنترل و نظارت بر شب در مدیریت امور شب خواهد بود. کاربران سیستم شامل متصدیان امور بانکی، رئیس بانک و کارمندان مدیریت امور شب، دارای تحصیلات تکمیلی هستند و دوره‌های ICDL را گذرانده‌اند. آن‌ها استفاده از صفحه کلید را به مoshواره ترجیح می‌دهند. هر یک از آن‌ها دارای کامپیوتری هستند که روی آن‌ها مرورگر اینترنت اکسپلورر نسخه ۷ یا بالاتر نصب شده است.

تا زمان استقرار سیستم، بانک دارای ۱۰۰ شعبه و ۱۰ مدیریت امور شب خواهد بود که در مجموع شامل ۱۰۰۰ متصدی امور بانکی، ۱۰۰ رئیس شعبه و ۵۰ نفر کارشناس مدیریت امور شب خواهند بود.»

۶- تعیین مرز سیستم

توافق بر سر مسأله و شناسایی ذینفعان و کاربران در مراحل قبلی انجام شد. حال نوبت به تعریف سیستمی رسیده است که انتظار می‌رود با ساخت و راهاندازی آن مسائل مرتفع و انتظارات برآورده گردد.

برای تعریف سیستم، تعیین مرز و توافق بر سر آن ضروری است. مرز سیستم خطی بین راه حل و دنیای واقعی اطراف آن است. به عبارتی مرز سیستم، ناحیه‌ای را مشخص می‌کند که راه حل در آن قرار می‌گیرد و از دنیای واقعی مجزا می‌گردد.



شکل ۳-۷: ورودی/سیستم/خروجی

سیستم، اطلاعات را در قالب ورودی از کاربران دریافت و پس از پردازش در قالب خروجی به آنان بازمی‌گرداند (شکل ۳-۷). رابطه‌ای^۱ سیستم پل ارتباطی آن با دنیای خارج و کاربران هستند. با این رویکرد دنیای ذهنی پیرامون سیستم به دو بخش زیر تقسیم می‌شود:

- داخل سیستم یا به اختصار سیستم
 - خارج سیستم یا آن‌هایی که با سیستم در تعامل هستند
- مرز سیستم حد فاصل داخل سیستم را از خارج آن مشخص می‌نماید. شناسایی مرز نه تنها در تدوین محدوده^۲ پروژه نقش بسزایی دارد بلکه راه حل را نیز بیان می‌کند. بسیاری از موقع تعیین مرزهای سیستم کار ساده‌ای است. برای مثال تعیین مرز سیستم دفترچه تلفن شخصی و تک کاربره کار دشواری نیست، اما شناسایی مرز سیستم پیچیده‌ای مانند «سیستم حساب سپرده کوتاه مدت» کار دشواری است. برای این کار لازم است به نمونه پرسش‌های زیر پاسخ داده شود:
- آیا سیستم تنها در شعبه مورد استفاده قرار خواهد گرفت یا در ستاد نیز به کار گرفته خواهد شد؟
 - آیا سیستم تنها برای عملیات بانکی مانند واریز و برداشت مورد استفاده قرار می‌گیرد یا برای سایر امور مانند عملیات بازرگانی، کنترل و نظارت نیز کاربرد خواهد داشت؟
- برای تعیین دقیق مرز سیستم علاوه بر تشریح مشخصات آن (داخل سیستم)،^۳ نمای بیرونی سیستم و

1. Interface
2. Scope

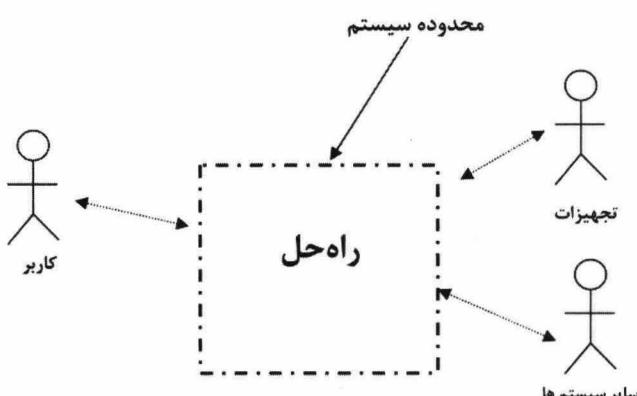
۳- روش تشریح مشخصات در فصل‌های بعدی ارائه شده است.

کسانی که با آن در تعاملند نیز شناسایی و مشخص می‌گردند. تعامل کنندگان می‌توانند کاربرانی مانند متصدی امور بانکی، تجهیزات سخت‌افزاری مانند بارکدخوان یا سیستم‌های دیگر مانند سیستم اطلاعات مشتریان باشند. تعامل کنندگان با سیستم، کنشگر^۱ نامیده می‌شوند. به عبارت دیگر «کنشگر کسی یا چیزی خارج از سیستم است که با آن در تعامل است»^۲. در UML کنشگر به شکل زیر نمایش داده می‌شود.



شکل ۳-۸: کنشگر

با تعریف مفهوم کنشگر مرز سیستم را می‌توان به شکل زیر نشان داد.



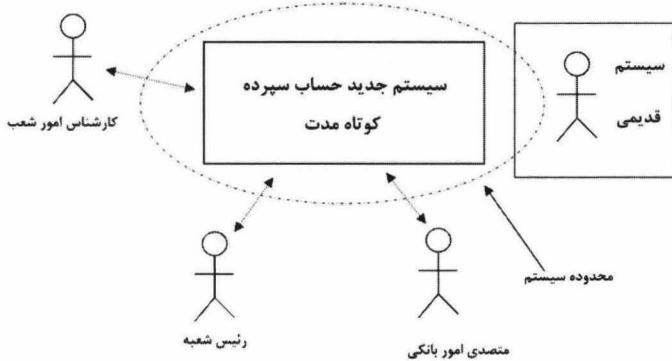
شکل ۳-۹: مرز سیستم

شناسایی کنشگرهای گام بسیار مهمی در تحلیل مساله است. تحلیل‌گر می‌تواند با شناسایی آن‌ها نمایی از سیستم را در قالب نموداری ترسیم کند که مرزهای سیستم، کنشگرهای سیستم و سایر رابطه‌های سیستم را نمایش می‌دهد.^۳ نمونه‌ای از آن برای «سیستم حساب سپرده کوتاه مدت» در زیر نمایش داده شده است.

۱. Actor

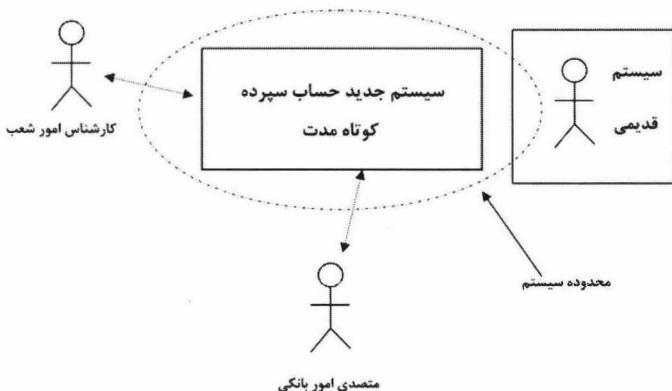
۲- کنشگرهای در فصل ۸ به تفصیل تشریح خواهند گردید.

۳- سلیقه تحلیل‌گر و فضای حاکم بر پروژه در ترسیم تصویر مؤثر است. در اینجا تنها نمونه‌ای از آن ارائه شده است.



شکل ۱۰-۳: نقش کنشگرها در تعیین مرز سیستم

برای تأکید بیشتر بر اهمیت مرز سیستم، نمونه دیگری از تصویر قبلی در شکل زیر نشان داده شده است:



شکل ۱۱-۳: نقش کنشگرها در تعیین مرز سیستم

با مقایسه دو تصویر مشخص می‌گردد که در تصویر دوم رئیس شعبه با سیستم تعاملی نخواهد داشت. به عبارت دیگر راه حل دوم به گونه‌ای ارائه شده است که رئیس شعبه امکان استفاده از سیستم را ندارد.

نمودار مذکور و توضیحات آن در بخش «دورنمای سیستم» در «سنند چشم‌انداز» تدوین می‌گردد^۱.

۱- تصویر ارائه شده تنها نمونه‌ای از تصویری است که می‌تواند در بخش «دورنمای سیستم» آورده شود.

۷- شناسایی قیدهای راه حل

قید به شکل زیر تعریف می‌گردد:

«محدودیت‌های حاکم بر درجه آزادی تیم توسعه در ارائه راه حل را قید گویند.» پیش از آغاز ارائه راه حل‌ها و تلاش برای ساخت سیستم باید به قیدهای موجود دقت کرد. هر قید می‌تواند به صورت بالقوه توانایی تیم توسعه در پیشنهاد و انتخاب راه حل را محدود سازد. از این رو لازم است شناسایی قیدها در فرآیند توسعه سیستم به عنوان بخش مهمی جای داده شود.

به عنوان مثال در توسعه سیستم حساب سپرده کوتاه مدت اگر قیدی وجود نداشته باشد تمامی راه حل‌های پیشنهادی قابل پذیرش خواهد بود. اما وجود قید، انتخاب و پذیرش راه حل را تحت تأثیر قرار خواهد داد. به عنوان مثال قید «لزوم ارتباط سیستم با سیستم اطلاعات مشتریان که حاوی اطلاعات مشتریان فعلی بانک است» راه حل پیشنهادی را در صورتی قابل قبول خواهد کرد که بتواند اطلاعات مشتریان فعلی را از سیستم مذکور دریافت نماید. به عنوان مثال دیگر قید «لزوم رعایت استانداردهای توسعه نرم افزار واحد فناوری اطلاعات» که در آن ذکر شده است «بستر نرم افزاری بانک مبتنی بر فناوری های شرکت مایکروسافت است» موجب پذیرفته شدن راه حل هایی می‌شود که بر اساس فناوری های شرکت مایکروسافت مانند SQL Server و .NET ارائه گردد.

برای شناسایی قیدها لازم است منابع مختلفی مانند زمانبندی، هزینه تجهیزات، عوامل محیطی، تکنولوژی ساخت و راه اندازی، مسابیل سیاسی و قانونی، خرید نرم افزارها، قوانین و روال های سازمانی و محدودیت های منابع انسانی بررسی گردند. گاه قیدها توسط مشتری بیان می‌گردند. به عنوان مثال «عدم تهیه سخت افزار جدید برای سیستم در حال توسعه و الزام به استفاده از سخت افزارهای موجود» در جلسات مصاحبه توسط واحد فناوری اطلاعات بانک به طور رسمی اعلام می‌گردد، اما تیم توسعه باید در قالب برنامه ریزی مشخص نسبت به استخراج قیدها اقدام کند. قیدهای حاکم بر راه حل در سندهای چشم انداز مدون می‌گردند.

در جدول صفحه بعد نمونه‌ای از منابع قیدها به همراه مجموعه پرسش‌هایی آورده شده است که پاسخ به آن‌ها منجر به شناسایی قیدها خواهد گردید.

جدول ۷-۳: منابع قیدهای پروژه

منابع	مثال
مسایل اقتصادی ^۱	آیا پروژه با محدودیت‌های مالی مواجه است؟ حق امتیاز محصول متعلق به چه کسی است؟
سیاست و قانون ^۲	آیا موارد سیاسی در سازمان وجود دارند که پروژه را تحت تأثیر قرار دهند؟ ^۳ آیا بین واحدهای سازمانی مرتبط با سیستم مسایلی وجود دارد که بر توسعه سیستم تأثیرگذار باشد؟
تکنولوژی	آیا برای انتخاب فناوری شرایطی وجود دارد؟ آیا فناوری سیستم باید مطابق سکو ^۴ و فناوری فعلی سازمان باشد؟ آیا می‌توان از تکنولوژی‌های جدید استفاده نمود؟
سیستم‌ها	آیا می‌توان در سیستم از قطعات نرم‌افزاری استفاده نمود که نیاز به خریداری داشته باشند؟ آیا سیستم جدید لازم است با سایر سیستم‌های موجود یکپارچه باشد؟ سیستم عامل و سایر ابزارهای مورد استفاده در سازمان چیست؟
محیط ^۵	آیا نیازمندی‌های امنیتی خاصی وجود دارد؟ آیا الزام به رعایت استانداردهای سازمانی خاصی وجود دارد؟ مانند استانداردهای ISO
زمانبندی و منابع ^۶	آیا پروژه دارای زمانبندی از پیش تعریف شده‌ای است؟ آیا نیازی به تأیید اعضای تیم توسعه توسط کارفرما وجود دارد؟

-۸- نکات کلیدی

- تحلیل مسئله، فرآیند شناسایی مشکلات دنیای واقعی و نیازهای کاربران و پیشنهاد راه حلی برای رفع آن‌ها است.
- هدف از تحلیل مسئله شناخت بهتر مسئله قبل از شروع ساخت سیستم است.
- لازم است کاربران و ذینفعان در شناسایی عوامل اصلی مسایل مشارکت نمایند.
- شناسایی کنشگرهای سیستم گامی مهم در تحلیل مسئله است.

1. Economics

2. Politics

۳- در اینجا سیاست به معنای مسایل حکومتی نیست. به عنوان مثال شکست پروژه مشابه در سازمان یکی از مسایل سیاسی است که لازم است تیم بدان توجه کند.

4. Platform

5 Environment

6 Schedule and resources

۹- مراجع

1. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
 2. Gause, Donald, and Gerald Weinberg. Exploring Requirements: Quality Before Design. Dorset House Publishing, 1989.
 3. Leffingwell, Dean ,Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
 4. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
 5. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>
۶. پولیا، جورج. چگونه مسأله را حل کنیم. با ترجمه احمد آرام. انتشارات کیهان، ۱۳۸۵.





بخش سوم: شناسایی نیازهای ذینفعان

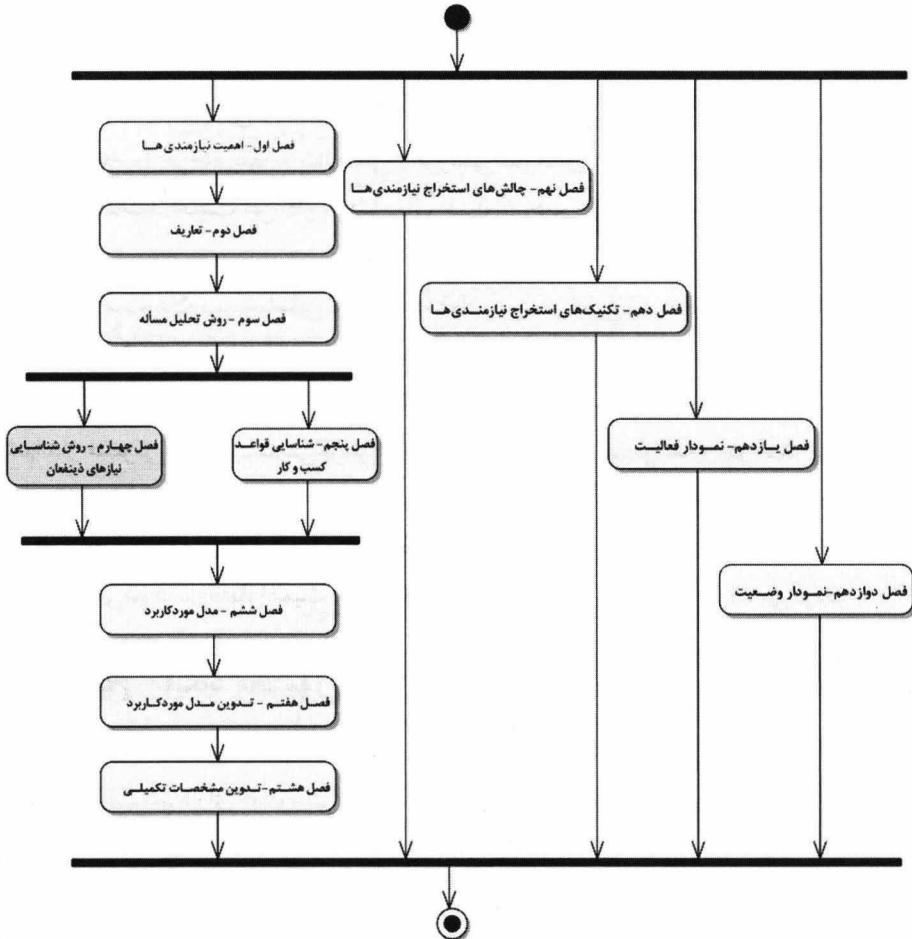
هدف این بخش، شناسایی نیازهای ذینفعان اصلی پژوهه در قالب جمع‌آوری اطلاعاتِ محصول مورد انتظار و مطلوب آن‌ها است. علاوه بر شناسایی نیازهای ذینفعان، قواعد حاکم بر کسب‌وکار به منظور استفاده در تعریف سیستم، استخراج و مدون می‌گردد.

فصل چهارم – روش شناسایی نیازهای ذینفعان
فصل پنجم – شناسایی قواعد کسب‌وکار

فصل چهارم - روش شناسایی نیازهای ذیفعان

اگر به جای استفاده از فرایندی کامل و کارا، تنها مجاز به تهیه یک مستند، مدل یا فراورده در پروژه توسعه نرم‌افزار بودم، سند چشم‌انداز مختصر و حرفه‌ای، تنها انتخاب من بود.

فیلیپ کراچن از مسئولین ارشد RUP



۱- مقدمه

مرحله اول پروژه به شناسایی و تحلیل مسأله اختصاص دارد که شناسایی ذینفعان را نیز شامل می‌شود (بخش دوم کتاب). مرحله بعد به شناسایی و بررسی نیازهای درخواستی ذینفعان اختصاص دارد (بخش سوم کتاب). اهمیت این کار به دلیل تأثیر فراوان رفع نیازها و درخواست‌های ذینفعان در موقوفیت پروژه است.

ضروری است علاوه بر شناسایی نیازهای ذینفعان، سیستم نهایی نیز به صورت کلان توصیف گردد تا امکان کسب توافق ذینفعان و تعیین انتظارات آن‌ها از سیستم برای تیم توسعه فراهم گردد. شناسایی نیازها و تعیین انتظارات ذینفعان به همراه توصیف کلان سیستم منجر به تدوین «جسم انداز مشترک»^۱ از سیستم می‌گردد که همویی ذینفعان و تیم توسعه را در ادامه پروژه به همراه خواهد داشت. «جسم انداز مشترک» به شکل سندی با نام «سندهای جسم انداز» تدوین می‌گردد. در این فصل، گام‌های شناسایی نیازهای ذینفعان، تعیین انتظارات آن‌ها و توصیف کلان سیستم توضیح داده می‌شود. در انتهای، اهمیت و جایگاه سندهای جسم انداز و قالب آن نیز ارائه می‌گردد.

۲- تشریح جایگاه محصول

آن چه که ذینفعان را مجبور به توسعه سیستم می‌کند، بازتابی از مشکلات یا فرصت‌های آنان است. به عبارت دیگر، سیستم دارای «مأموریتی»^۲ است که ساختش را توجیه می‌کند. تیم توسعه باید «مأموریت» سیستم را با عبارات ساده و کوتاه در «سندهای جسم انداز» بیان نماید.

فرض کنید مدیر ارشد کارفرما (بالاترین مقام تصمیم‌گیری و با نفوذترین ذینفع پروژه که معمولاً از جزئیات توسعه سیستم‌ها مطلع نیست) را در آسانسور ملاقات می‌نماید. وی از شما می‌پرسد «سیستمی که در حال حاضر مشغول توسعه آن هستید، چیست و چه کاربردی دارد؟». روشن است که پاسخ‌گویی به ایشان تا چه اندازه اهمیت دارد. از آنجا که وی به زودی آسانسور را ترک خواهد کرد، فرصت کوتاهی برای پاسخ‌گویی دارد. ناچارید جایگاه سیستم و ارزش آن را برای ذینفعان، خلاصه و سریع، به گونه‌ای بیان کنید که وی توجیه و مجاب شود. پاسخ شما، متنی است که در قسمت «شرح جایگاه محصول» در سندهای جسم انداز باید آورده شود.

هدف از تدوین این قسمت، ارائه تعریف ساده و کاملی از سیستم و کاربردهایش به ذینفعان است. اگر تیم توسعه قادر به تکمیل این قسمت نباشد، می‌توان نتیجه گرفت که برداشتِ درست و دقیقی از کاربرد سیستم ندارد.

جایگاه محصول در قالب جدول زیر مدون می‌گردد.

جدول ۴-۱: قالب شرح جایگاه محصول

برای	[نام کاربران]
که نیاز به	[شرح نیازهای کاربران(مشکلات و فرصت‌ها)]
[نام محصول / سیستم]	که یک سیستم [دسته‌بندی محصول] است
قادر به	[توانایی‌های اصلی سیستم]
که بر خلاف	[انتخاب جایگزین یا وضعیت فعلی]
قادر به	[شرح خلاصه‌ای از تفاوت‌های سیستم با انتخاب جایگزین / وضعیت فعلی]

جدول بالا عبارت زیر است که به شکل جدول مدون گردیده است:
«برای [...] که نیاز به [...، ...] که یک سیستم [...] است، قادر به [...] که بر خلاف [...]، قادر به [...]».

جدول ۴-۲: نمونه شرح جایگاه محصول

برای	[متصدیان امور بانکی، روسای شعب و کارشناسان امور شعب
که نیاز به	ارائه خدمات بانکی به مشتریان، مدیریت فعالیت‌های شعبه و مدیریت فعالیت‌های ستادی دارند
سیستم حساب سپرده کوتاه مدت	که یک سیستم اطلاعاتی است
قادر به	انجام خدمات بانکی در شعب، کنترل فعالیت‌های متصدیان امور بانکی، کنترل قواعد و قوانین بانکی است
که بر خلاف	روش غیرمکانیزه فعلی
قادر به	ارائه خدمات بانکی به مشتریان در تمامی شعب علاوه بر شعبه افتتاح کننده، ارائه آخرین اطلاعات مشتریان و حساب‌ها و کنترل نقدینگی شعب است.

۳- تدوین نیازهای اصلی ذینفعان و کاربران

شناسایی نیازها یکی از مهم‌ترین و پیچیده‌ترین کارهای تیم توسعه است. یکی از دلایل پیچیدگی شناسایی نیازها، بیان چگونگی رفع نیازها در سیستم (راه حل) به جای اعلام نیازهای واقعی (مسئله) توسط ذینفعان است. ذینفعان، نیاز واقعی خود را به رفتارهایی در سیستم ترجمه می‌نمایند که به عقیده آن‌ها، به درستی نیاز را رفع می‌نماید. آنگاه رفتارهای مذکور را به عنوان نیاز واقعی به تیم توسعه اعلام می‌کنند. این معضل با افزایش تجربه ذینفعان در توسعه سیستم‌های نرم‌افزاری، تشدید می‌گردد. به مثال‌های جدول ۳-۴ توجه نمایید.

جدول ۴-۳: تفاوت نیاز واقعی و نیاز اعلام شده

نیاز واقعی	نیاز اعلام شده توسط ذینفع
انجام عملیات بانکی با حساب مشتری در کلیه شعب، علاوه بر شعبه افتتاح‌کننده ممکن باشد.	سیستم باید بانک اطلاعاتی متمرکز داشته باشد.
طبق ضوابط شورای عالی پول و اعتبار، ضروری است واریزکننده احراز هویت گردد.	یک حوزه اطلاعاتی (فیدی) روی صفحه افتتاح حساب وجود داشته باشد که کد ملی واریزکننده در آن وارد شود.

اگر ذینفعان، دانش و تجربه کافی در حوزه مسئله^۱ و توسعه سیستم‌های نرم‌افزاری داشته باشند، این کار نه تنها ایرادی ندارد، بلکه توصیه هم می‌گردد اما در شرایطی که ذینفعان، صلاحیت لازم (شناخت کل حوزه مسئله و تجربه کافی در توسعه سیستم نرم‌افزاری) را نداشته باشند، این موضوع مشکل‌زا می‌گردد. در چنین شرایطی، آگاهی دادن به ذینفعان (لزوم بیان نیازهای واقعی) در کاهش معضل، کارساز است. به علاوه، تیم توسعه باید با به کارگیری تکنیک‌های مناسب، نیازهای واقعی را از نیازهای غیرواقعی جدا کند.

همان‌طور که گفته شد، شناسایی نیازها یکی از مهم‌ترین کارهای تیم توسعه است. شناسایی نادرست نیازهای واقعی یا بی‌اطلاعی از آن‌ها، یکی از ریسک‌های مهم پروژه‌های توسعه نرم‌افزار است. اگر تیم توسعه در شناسایی نیازها، تنها به نیازهای اعلام‌شده ذینفعان بسته نماید، با وجود مجموعه‌ای از امکانات در سیستم نهایی، احتمال تحقق نیافتی اهداف اصلی پروژه، زیاد است. وقوع چنین شرایطی به معنای شکست پروژه است.^۲

کسب رضایت تمامی ذینفعان، موضوع چالش برانگیزی است. ذینفعان انتظار دارند که نیازهای متفاوت و گاه متناقض آن‌ها، در پروژه رفع گردد. اغلب در این شرایط جمع‌آوری نیازها و کسب توافق ذینفعان به بن‌بست می‌رسد. استفاده از روش‌های اولویت‌بندی و امتیازدهی در رفع تناقض نیازها و اختلاف بین ذینفعان، مؤثر است. در این روش، ذینفعان، نیازها را امتیازدهی و انتخاب می‌نمایند. نیازهای دارای امتیاز بیشتر، پذیرفته می‌شوند و در دستور کار قرار می‌گیرند.

بخشی از نیازهای واقعی ذینفعان، تنگناها و کاستی‌های وضعیت جاری است. تیم توسعه باید از فرایندها و وضعیت جاری سازمان ذینفع، اطلاعات کافی کسب نماید و به کمک تکنیک‌های تحلیل و اظهارات ذینفعان، تنگناهای وضعیت جاری (نیازهای واقعی) را شناسایی نماید. در این راستا، شناخت، مستندسازی و مدل‌سازی فرایند کسب‌وکار جاری انجام می‌گردد. اغلب، این مدل‌ها و مستندات به

۱- اغلب کاربران و ذینفعان بر بخشی از حوزه مسئله مسلط و از تأثیرات راه حل‌های خود بر سایر بخش‌ها ناگاه هستند.
۲- به تحلیل گر توصیه می‌شود تا قبل از احراز درستی شرایط ذیل، تحلیل نیازمندی‌ها را ادامه ندهد: الف: صلاحیت اعلام‌کننده نیاز ب: واقعی بودن نیاز.

تأثید ذینفع نیز می‌رسد.^۱ بررسی فرایندها و شناسایی نیازها می‌تواند با برگزاری کارگاه نیازمندی‌ها^۲ انجام شود.

نیازهای اصلی در قسمت «نیازهای اصلی ذینفعان و کاربران» سند چشم‌انداز و به شکل جدول زیر تدوین می‌گردند.

جدول ۴-۴: روش تعریف نیازها

نیاز	اولویت	دلیل اهمیت	راه حل فعلی	راه حل پیشنهادی

شرح هر یک از ستون‌ها در زیر آمده است.

- ﴿ نیاز: عنوان یا شرح نیاز
- ﴿ اولویت^۳: درجه اهمیت نیاز برای ذینفع که می‌تواند یکی از مقادیر بالا، متوسط و پایین^۴ باشد.^۵
- ﴿ دلیل اهمیت^۶: دلیل اهمیت نیاز برای ذینفع
- ﴿ راه حل فعلی: روش فعلی رفع نیاز توسط ذینفعان
- ﴿ راه حل پیشنهادی: روش پیشنهادی تیم توسعه برای رفع نیاز با تکیه بر امکانات سیستم به عنوان مثال به نمونه‌های زیر در سیستم «حساب سپرده کوتاه مدت» توجه نمایید.

جدول ۴-۵: نمونه نیازهای سیستم حساب کوتاه مدت

نیاز	دلیل اهمیت	راه حل فعلی	راه حل پیشنهادی
ارائه خدمات بانکی به مشتریان در کلیه شعب علاوه بر شعبه افتتاح‌کننده حساب	کسب رضایت مشتریان و مزیت رقابتی در بازار	وجود ندارد.	ارائه کلیه خدمات بانکی از تمامی شعب با سیستم امکان‌پذیر خواهد بود.
بررسی پرونده مشتریان	برای اعطای تسهیلات استخراج می‌گردد.	حساب مشتریان فایلهای اکسل، گردش حساب های مشتری را گزارش می‌دهد.	با انتخاب مشتری و بازه زمانی، سیستم خلاصه‌ای از گردش حساب های مشتری را گزارش می‌دهد.

۱- مستندسازی و مدل‌سازی وضعیت جاری، به‌خصوص فرایندهای کسب‌وکار در نظام «مدل‌سازی کسب‌وکار» RUP قرار دارد که در این کتاب آورده نشده است.

۲- کارگاه نیازمندی‌ها، یکی از تکنیک‌های استخراج نیازمندی‌ها است که در فصل دهم تشریح شده است.

۳- درجه اهمیت نیازها به هدف‌گذاری و برنامه‌ریزی مناسب‌تر پروژه، کمک می‌کند.

4. High, Medium, Low

۵- مقادیر اشاره شده، مقادیر پیشنهادی است.

6. Concern

در این قسمت از سند چشم‌انداز، خلاصه‌ای از مهم‌ترین نیازهای ذینفعان و کاربران نوشته می‌شود. معمولاً نیازهای این جدول، دارای اولویت «بالا» یا «متوسط» هستند. توصیه می‌گردد که تعداد نیازهای این جدول، کمتر از هفت مورد باشد. اگر تدوین تمامی نیازها لازم باشد، تحلیل‌گر می‌تواند مستند جدأگاههای برای آن‌ها تهیه نماید.^۱

۴- شناسایی ویژگی‌های سیستم

بعد از شناسایی نیازها، تیم توسعه به دنبال راه حلی مبتنی بر نرم‌افزار خواهد بود تا بتواند نیازهای ذینفعان را رفع نماید. برای بیان این راه حل از «ویژگی‌ها» استفاده می‌شود. همان‌طور که در فصل دوم بیان شد «ویژگی، خدمتی است که توسط سیستم برای پوشش نیازهای ذینفعان ارائه می‌شود». ویژگی‌ها در فضای راه حل جای دارند، در حالی که نیازها از حوزه مسأله می‌آیند.

نوع، اهمیت و چگونگی بیان ویژگی‌ها متفاوت است. به نمونه ویژگی‌های زیر توجه نمایید:

- ویژگی ۱: برداشت از حساب مشتری در کلیه شعب امکان‌پذیر باشد.
 - ویژگی ۲: امکان افتتاح بیش از یک حساب کوتاه مدت برای مشتری در هر شعبه وجود داشته باشد.
 - ویژگی ۳: اطلاعات مشتریان در سیستم نگهداری گردد.
 - ویژگی ۴: نگهداری سوابق تغییرات موجودی حساب مشتری الزامی است.
 - ویژگی ۵: عملیات انجام شده توسط کاربر ثبت گردد.
- نمونه ویژگی‌های ارائه شده، در سطوح متفاوتی از جزئیات، بیان شده‌اند. ویژگی اول، دوم و پنجم بیان‌گر سرویس‌های کلان و بسیار بالاهمیت سیستم هستند. ویژگی سوم خصوصیات دو مورد اول (کلان و بسیار بالاهمیت) را ندارد (اطلاعات مشتریان به دلیل امکان انجام خدمات شعبه‌ای مانند واریز و برداشت در سیستم ثبت می‌گردد و به تنها بی ارزشی برای کاربر ایجاد نمی‌کند). ویژگی چهارم، جزئی تر از موارد قبلی است.

سه ویژگی اول، وظیفه‌مندی سیستم هستند، در حالی که ویژگی چهارم و پنجم، مشخصات کیفی سیستم را بیان می‌نماید که تاریخچه تغییرات^۲ و رویدادنگاری^۳ نامیده می‌شود و از نیازمندی‌های غیروظیفه‌مندی سیستم هستند.

ویژگی‌های اول و دوم، ریشه در حوزه مسأله دارند:

- ویژگی ۱ ناشی از فرصت است:

۱- این مستند «خواسته‌های ذینفعان» (Stakeholder Requests) نامیده می‌شود.

بانک به دنبال کسب جایگاه مناسبی در بازار رقابتی است و باید خدمات مناسب تری به مشتریان ارائه دهد. لذا خدمات بانکی باید در تمامی شعبه به مشتری ارائه گردد.

○ ویژگی ۲ ناشی از مشکل است:

عدم امکان افتتاح بیش از یک حساب برای مشتری در یک شعبه، به دلیل وجود محدودیت های سازمانی در بانک است، که بانک قصد دارد با استقرار سیستم، آن را مرتفع نماید.^۱

سه ویژگی آخر، ریشه در حوزه راه حل مبتنی بر نرم افزار دارند. هر چند که نگهداری اطلاعات مشتریان به تنها ی ارزشی برای ذینفع ایجاد نمی کند، اما پیش نیاز افتتاح حساب در نرم افزار است. وجود ویژگی های چهارم و پنجم، به منظور کنترل و رفع اشتباہات کاربری است.

گاه خدمت یا خصوصیتی از سیستم برای ذینفع، اهمیت ویژه ای دارد و لازم می داند که مورد مذکور در ویژگی ها مدون گردد. با وجود آن که ممکن است این مورد، حائز شرایط تدوین در قسمت ویژگی ها نباشد (کلان و بالاهمیت) تیم توسعه برای کسب توافق ذینفع، به تاچار آن را در فهرست ویژگی ها مدون می نماید. چگونگی تدوین ویژگی ها و کسب توافق ذینفعان، کار بسیار دشواری است.

۴- روش های شناسایی ویژگی ها

ویژگی های سیستم به روش های زیر شناسایی می گردند:

- تحلیل چگونگی رفع نیازهای ذینفعان
- تحلیل راه کار پیشنهادی به کسب و کار
- تحلیل نیازمندی های نرم افزاری

تحلیل چگونگی رفع نیازهای ذینفعان^۲

در این روش، چگونگی رفع نیازهای ذینفعان در سیستم، تحلیل می شود. اگر مجموعه ویژگی های شناسایی شده، برخی از نیازها رفع ننماید، لازم است ویژگی های جدیدی به سیستم اضافه شود تا نیازهای مذکور بر طرف گرددند. به عنوان مثال در سیستم حساب سپرده کوتاه مدت، ذینفع انتظار دارد تا نقدینگی شعبه در مدیریت امور شعب، قابل کنترل باشد (نیاز: کنترل نقدینگی شعبه). برای رفع این نیاز، سیستم باید در هر لحظه بتواند موجودی شعبه را محاسبه نماید. برای این کار، موجودی اولیه شعبه^۳، اسکناس های ورودی به شعبه و خروجی از آن در هر روز باید ثبت گردد. لذا سیستم باید

۱- در بانک ها، چنین محدودیتی وجود ندارد و مشکل عینیت خارجی ندارد.

۲- یادآوری می گردد که در پروژه های پیمانکاری، ذینفعان شامل واحد های پشتیبانی، استقرار و بازاریابی و فروش پیمانکار نیز هستند. توجه به این دسته از ذینفعان بخصوص واحد بازاریابی و فروش زمانی اهمیت بیشتری پیدا می کند که پیمانکار با اجرای پروژه، به دنبال تهیه بسته نرم افزاری (Package) است.

۳- موجودی اولیه شعبه به معنای موجودی شعبه در هنگام اولین استفاده از سیستم است.

ویژگی‌های زیر را داشته باشد:

- ویژگی ۱: تعریف شعبه (امکان تعیین موجودی اولیه وجود داشته باشد)
- ویژگی ۲: واریز به حساب
- ویژگی ۳: برداشت از حساب
- ویژگی ۴: تحويل اسکناس به مدیریت شعب
- ویژگی ۵: دریافت اسکناس از مدیریت شعب
- ویژگی ۶: گزارش موجودی شعبه

اگر سیستم قادر به پوشش حتی یکی از ویژگی‌های مذکور نباشد، رفع نیاز «کترل نقدینگی شعبه» میسر نیست.

این روش می‌تواند در حین یا پایان شناسایی ویژگی‌ها، با هدف بررسی پوشایش بودن ویژگی‌ها نسبت به نیازها به کار گرفته شود.

تحلیل راه کار پیشنهادی به کسب و کار

در این روش، فرایند کسب و کار پیشنهادی، بررسی و تحلیل می‌گردد و امکانات کلان مورد نیاز برای پشتیبانی فرایند به عنوان ویژگی شناسایی می‌گردد. به مثال زیر توجه نمایید:

فرایند پیشنهادی «واریز به حساب»:

- مشتری فرم واریز را تکمیل می‌کند و به همراه وجه به متصدی امور بانکی تحويل می‌دهد.
- متصدی امور بانکی فرم واریز را در سیستم ثبت می‌کند. سیستم علاوه بر ثبت اطلاعات فرم واریز، استناد حسابداری آن را نیز صادر می‌کند.
- متصدی امور بانکی به کمک سیستم، فرم را پرفرماز می‌کند و پس از مهر و امضاء، یک نسخه آن را به مشتری تحويل می‌دهد.
- ○

در اینجا ویژگی‌های مورد انتظار از سیستم عبارتند از:

- ویژگی ۱: واریز به حساب
 - ویژگی ۲: ماشین کردن فرم واریز
 - ویژگی ۳: صدور سند حسابداری مربوط به واریز به حساب
- پس از تحلیل تمامی فرایندها، ویژگی‌های شناسایی شده پالایش می‌گردند و مجموعه نهایی به عنوان ویژگی‌های سیستم تدوین می‌گردد.

تحلیل نیازمندی‌های نرم‌افزاری

شناسایی مورد کاربردها و تدوین مشخصات آن‌ها (به فصل شش و هفت مراجعه شود)، گام بعدی

تحلیل است. در آن گام، اغلب راه حل های پیشنهادی، مستلزم وجود ویژگی هایی هستند که قبل از شناسایی نشده‌اند. به عبارت دیگر، ماهیت رفت و برگشتی بین این دو سطح از نیازمندی‌ها (ویژگی و نیازمندی نرم‌افزاری) وجود دارد. به عنوان مثال در تدوین مورد کاربرد «ورود به سیستم»، پرسش زیر مطرح می‌گردد:

«پس از ورود متصلی یا رئیس بانک به سیستم، شعبه کاری آن‌ها چگونه شناسایی می‌گردد؟»

پاسخ به پرسش مذکور، منجر به اضافه شدن ویژگی زیر خواهد شد:

«امکان تعیین پرسنل هر شعبه و سمت آن‌ها وجود داشته باشد.»

پرسش دیگری نیز در همان مورد کاربرد مطرح می‌گردد:

«سیستم چگونه استفاده کاربر از سیستم را به روزهای کاری محدود می‌کند؟ باید به یاد داشت گاهی روزهای تعطیل رسمی مانند جمعه‌های پایان سال نیز، روز کاری هستند.»

پاسخ به این پرسش نیز منجر به اضافه شدن ویژگی «تقویم کاری» می‌شود تا تعیین روزهای کاری و غیرکاری ممکن گردد.

۴- مستندسازی ویژگی‌ها

ویژگی‌ها در قسمت «ویژگی‌های محصول» سند چشم‌انداز مستند می‌گردند. مستندسازی هر ویژگی، شامل نام و شرح آن است. به نمونه زیر از سیستم حساب سپرده کوتاه مدت توجه نمایید:

۱- محاسبه سود

سیستم امکان محاسبه سود و واریز آن‌ها به حساب‌های مشتریان را خواهد داشت. اسناد حسابداری مربوط به واریز سود نیز صادر خواهد گردید.

۲- واریز به حساب مشتری

واریز مبالغ نقدی به حساب‌های مشتریان در سیستم مقدور خواهد بود.

با افزایش تعداد ویژگی‌ها، گروه‌بندی آن‌ها امری ضروری است. توصیه می‌گردد که گروه‌ها، عنوانی از حوزه مسئله یا راه حل باشند. نمونه‌ای از دسته‌بندی ویژگی‌ها در سیستم حساب سپرده کوتاه مدت، در ادامه آورده شده است.

○ مدیریت شب و پرسنل

شامل ویژگی‌هایی مانند تعریف شب و تعریف پرسنل است که توسط مدیریت امور شب استفاده می‌گردد.

○ خدمات به مشتری

شامل ویژگی‌هایی مانند تعریف مشتری، واریز به حساب و برداشت از حساب است که مرتبط با خدماتی است که بانک به مشتری ارائه می‌دهد.

○ مدیریت کاربران

شامل ویژگی‌هایی است که مرتبط با تعریف کاربران، نقش‌ها و حقوق دسترسی آنان است.

جدول ۴-۶: دسته‌بندی ویژگی‌ها در سند چشم‌انداز

.....

5- ویژگی های سیستم

5-1- مدیریت شعب و پرسنل

5-1-1- تعریف شعبه

کاربر می تواند شعبه جدیدی را در سیستم تعریف نماید یا اطلاعات شعب قبلی را تغییر دهد.

.....

5-2- خدمات به مشتری

5-2-1- واریز به حساب

واریز مبالغ نقدی به حساب های مشتریان در سیستم مقدور خواهد بود.

.....

در جدول ۴-۶، تأثیر دسته‌بندی ویژگی‌ها در سند چشم‌انداز نشان داده شده است.

۴-۳- تعیین ترتیب و اولویت بندی ویژگی‌ها

در این گام، اولویت ویژگی‌ها تعیین می‌گردد. اولویت‌بندی بخشی از مدیریت محدوده سیستم است. در این مرحله، اولویت‌بندی تنها از دیدگاه ذینفعان انجام می‌گردد.^۱ پس از تعیین اولویت ویژگی‌ها، می‌توان آن‌ها را در جدولی به شکل زیر در قسمت «ترتیب و اولویت‌بندی» در سند چشم‌انداز ارائه داد.^۲

حدول ۴-۷: اولویت سندی و پیشگم‌ها

ردیف	عنوان ویژگی	اولویت	نسخه

۱- به دلیل گستردگی موضوع و ارتباط آن با مدیریت محدوده، از توضیح روش‌ها و تکنیک‌های اولویت‌بندی و متبازان، صفت‌نظر شده است.

-۲- می توان از روش های دیگری مانند استفاده از ابزار های مدیریت نیازمندی ها یا ارائه اولویت بندی در فایل اکسل نیز استفاده نمود. گاهی دیده شده است که قسمت «ویژگی های سیستم» و قسمت «ترتیب و اولویت بندی» در قالب یک جدول ارائه شده است. برای پروژه های متوسط به بالا، روش مذکور توصیه نمی کردد، چرا که دسته بندی ویژگی ها در «ویژگی های سیستم» با «ترتیب» و «اولویت بندی» آنها متفاوت است. برخلاف قسمت «ویژگی های سیستم» در این قسمت، ویژگی ها بر اساس شماره نسخه یا اولویت، مرتب می گردند.

شرح هر یک از ستون‌ها در زیر آمده است.

- » عنوان ویژگی: عنوان ویژگی که در قسمت ویژگی‌های سند چشم‌انداز آمده است.
- » اولویت: یکی از عبارات «بالا»، «متوسط» و «پایین» که نشانگر اولویت ویژگی است.
- » نسخه: در صورتی که سیستم در بیش از یک نسخه تحویل داده می‌شود، شماره نسخه‌ای است که ویژگی در آن پیاده‌سازی می‌گردد.
- » به عنوان مثال، جدول زیر از سیستم حساب کوتاه مدت انتخاب شده است.

جدول ۴-۸: نمونه ویژگی‌های اولویت‌بندی شده در سیستم حساب کوتاه مدت

ردیف	عنوان ویژگی	اولویت
۱	اطلاعات مشتریان	بالا
۲	افتتاح حساب مشتری	بالا
۳	واریز به حساب	بالا
۴	برداشت از حساب	بالا
...	
۲۰	ارسال گزارش عملکرد سه ماهه شعبه از طریق پست الکترونیکی به واحد آمار	پایین

۵- توصیف کلان محصول

در این مرحله، قابلیت‌های کلان سیستم توصیف می‌شوند که در قسمت «مرور محصول» سند چشم‌انداز مدون می‌گردند. این قسمت شامل زیرقسمت‌های «دورنمای محصول» و «فرضیات و وابستگی‌ها» است.

۵-۱- تشریح دورنمای محصول

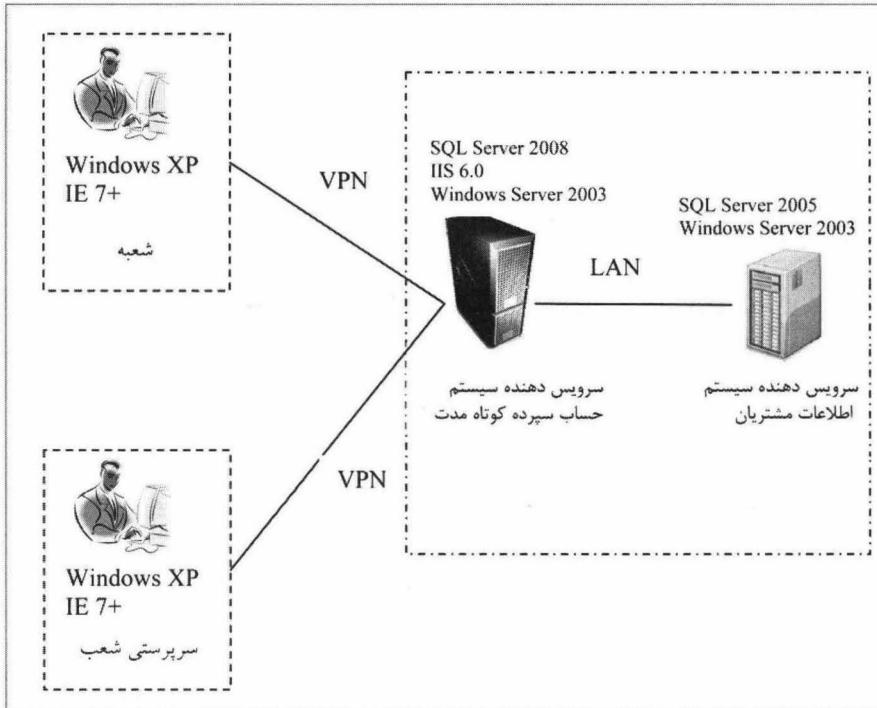
هدف از تشریح «دورنمای محصول»، تشریح رابطه سیستم با سیستم‌های موجود در محیط کاربران است. اگر نیاز به ارتباط با سیستم‌های دیگر باشد، چگونگی ارتباط و واسطه‌های ارتباطی آن‌ها، مشخص می‌گردد. چنانچه سیستم، بخشی از سیستم بزرگتری باشد، لازم است جایگاه آن در سیستم بزرگتر و ارتباط آن با زیرسیستم‌های دیگر مشخص گردد.

برای این کار توصیه می‌شود از نمودار بلوکی^۱ استفاده گردد. در این نمودار، سیستم‌های دیگر، ارتباطات آن‌ها و واسطه‌های لازم برای ارتباط، ترسیم می‌گردد. برای نمایش ارتباط داده‌ای بین سیستم‌ها، استفاده از نمودار متن^۲ توصیه می‌گردد.

شکل زیر از «دورنمای محصول» سیستم حساب سپرده کوتاه مدت انتخاب شده است. این تصویر

1. Block diagram
2. Context diagram

نشان می‌دهد که این سیستم با سیستم «اطلاعات مشتریان» در ارتباط است. بانک اطلاعاتی سیستم اطلاعات مشتریان، SQL Server 2005 است. این بانک اطلاعاتی در سرویس‌دهنده‌ای با سیستم عامل ویندوز ۳، ۲۰۰۳، قرار دارد.



شکل ۴-۱: نمای سیستم حساب سپرده کوتاه مدت

۵-۲-۵- تدوین مفروضات و وابستگی‌ها

در این مرحله، مفروضاتی تشریح می‌گردد که در صورت تغییر آن‌ها، سند چشم‌انداز بی‌اعتبار خواهد شد. تغییر مفروضات، نیازمند توافق مجدد طرفین خواهد بود. در این قسمت، اهم وابستگی‌های سیستم به سایر سیستم‌ها یا محیط کاربران، مدون می‌گردد. به عنوان مثال در سیستم حساب سپرده کوتاه مدت، فرض‌های زیر در توسعه سیستم لحاظ شده است:

«کارفرما حداقل تا سه سال آینده، سیستم اطلاعات مشتریان را استفاده خواهد نمود و تغییرات آتی آن بر ارتباط دو سیستم تأثیرگذار نخواهد بود. در صورت تأثیرگذاری، کسب توافق پیمانکار الزامی است».

۶- شناسایی نیازمندی‌های مستندسازی

این مرحله منجر به تعیین مجموعه‌ای از مستندات می‌گردد که به همراه محصول به ذینفعان، تحویل داده می‌شوند. مستندات مذکور با مستندات و مدل‌های فرایندی^۱ متفاوت است.^۲ به عنوان مثال مستندات زیر در سیستم حساب کوتاه مدت به همراه محصول تحویل می‌گردد:

- راهنمای برخط^۳

برای راهنمایی کاربران در هنگام استفاده، سیستم دارای راهنمای برخط است. راهنمای برخط در قالب صفحات HTML و دارای فهرست مطالب، امکانات جستجو و نمایه^۴ است.

- راهنمای نصب سیستم

راهنمای نصب سیستم برای راهبر سیستم تهیه می‌گردد و شامل موارد زیر است:

- حداقل مشخصات سخت‌افزاری
- گام‌ها و دستورات نصب
- خطاهای رفع شده و ویژگی‌های اضافه شده در هر نسخه

نیازمندی‌های مستندسازی در قسمتی به همین نام در سند چشم‌انداز مدون می‌گردند.

۷- شناسایی سایر نیازمندی‌های محصول

برخی از نیازمندی‌های غیرکارکردی کلان و بسیار مهم باید به تأیید ذینفعان برسد. نیازمندی‌های مذکور در قسمت «سایر نیازمندی‌های محصول» از سند چشم‌انداز مدون می‌شوند. یادآوری می‌گردد که نیازمندی‌های غیرکارکردی عمومی سیستم در مستند «مشخصات تکمیلی» مدون می‌شوند.^۵

استانداردهای مورد استفاده، نیازمندی‌های سخت‌افزاری، زیرساختی، کارایی و محیطی از جمله مواردی هستند که در این قسمت بررسی می‌شوند. ممکن است برخی از نیازمندی‌های مذکور، به صورت ویژگی‌ها، در بخش «ویژگی‌های محصول» مدون شده باشند و نیازی به ذکر مجدد آن‌ها نباشد. نمونه‌های زیر از سیستم حساب کوتاه مدت، انتخاب شده است:

- ورود کاربران به سیستم با قرارداد SSL انجام شود.
- ۱۰۰۰ کاربر (با توجه به تعداد کارکنان شعب) بتوانند همزمان از سیستم استفاده نمایند.

۱- به تمام مواردی که در فرایند توسعه نرم‌افزار تهیه می‌گردند، محصولات کاری یا work products گفته می‌شود.

محصولات کاری شامل کدهای برنامه، مدل‌ها، مستندات و راهنمایها نیز می‌گردند.

۲- می‌توان به سند چشم‌انداز، طرح مدیریت پروژه و مدل طراحی بانک اطلاعاتی اشاره نمود.

3. Online help
4. Index

۵- به فصل هشتم مراجعه گردد

-۸ سند چشم‌انداز

مهم‌ترین مستند خروجی این مرحله از پروژه، سند چشم‌انداز سیستم است. در ادامه جایگاه، اهمیت و اجزای آن تشریح شده است. در انتها نیز توصیه‌هایی برای تدوین آن ارائه شده است.

-۱-۸ جایگاه سند چشم‌انداز

هدف از تهیه سند چشم‌انداز، مستندسازی اهداف، انگیزه‌ها و نیازهای ذینفعان، بازار هدف، محیط کاربران، سکوی^۱ نهایی و ویژگی‌های محصول در حال ساخت است. این مستند ابزاری برای تبادل نظرات و کسب توافق درباره «چرایی و چیستی» پروژه است.

این مستند، محصول را از دیدگاه ذینفعان، بر اساس نیازها و ویژگی‌های مورد درخواست آن‌ها تعریف می‌کند. از این‌رو، پس از تأیید ذینفعان، سند چشم‌انداز مبنایی برای تشریح جزئیات نیازمندی‌ها توسعه تیم توسعه خواهد بود.

سند چشم‌انداز اصلی‌ترین بخش‌های راه حل ارائه شده توسط تیم توسعه را به شکل قابل استنباط برای ذینفعان ارائه می‌کند. لذا مبنایی برای کنترل و تأیید پیشرفت پروژه و معیاری برای سنجش تصمیمات آتی پروژه است.

ذینفعان باید چشم‌انداز را بازنگری کنند و نظرات خود را رسماً اعلام نمایند. تأیید سند چشم‌انداز توسط ذینفعان، ضروری است. دست‌یابی به سند چشم‌انداز مورد توافق ذینفعان، از مسئولیت‌های مهم و بسیار دشوار تیم توسعه است. اگر سند چشم‌انداز توافق‌شده‌ای وجود نداشته باشد، کارهای زیر

غیرقابل انجام یا به سختی قابل انجام است:

- مشارکت مؤثر ذینفعان در پروژه
- اندازه‌گیری پیشرفت واقعی پروژه
- مدیریت محدوده پروژه
- سنجش تصمیمات آتی در پروژه
- توجیه نمایندگان جدید ذینفعان یا توسعه‌دهندگان تازه وارد به پروژه

همان طور که گفته شد، سند چشم‌انداز مبنای ادامه کار تحلیل و تشریح نیازمندی‌ها توسط تیم توسعه است. هم‌چنین، توصیف کلی سیستم مورد انتظار ذینفعان در این سند ذکر شده است. تیم توسعه، اصلی‌ترین بخش‌های راه حل را به گونه‌ای قابل استنباط برای ذینفعان، در این سند مدون می‌کند. از این‌رو، سند چشم‌انداز مانند کتابچه راهنمایی است که گروه‌های مشارکت‌کننده در پروژه، همواره به آن استناد می‌کنند.

سند چشم‌انداز «مستند نیازمندی‌های محصول»^۲ نیز خوانده می‌شود.

1. Platform

2. Product requirements document

۲-۸- اجزای سند چشم‌انداز

همان طور که گفته شد، «سند چشم‌انداز» حاوی توصیف محصول از دیدگاه ذینفعان است. فارغ از شکل و قالب، این سند موارد زیر را برای ذینفعان مشخص می‌نماید:

- مزایای ساخت سیستم و پیش‌بینی فرصت‌های پس از آن
- مسئله‌ای که سیستم حل خواهد کرد
- کاربران نهایی
- توصیف کلان و ظایف سیستم
- مهمترین و کلان‌ترین نیازمندی‌های غیرکارکرده سیستم

اجزای سند چشم‌انداز به گونه‌ای انتخاب شده‌اند که در مجموع، موارد بالا را برای ذینفعان مشخص نمایند. اجزای سند چشم‌انداز در ادامه آورده شده‌اند.

۱- مقدمه

در این قسمت از مستند مقدمه‌ای بر مستند نوشه می‌شود. مقدمه در قالب چهار بخش هدف، محدوده، مراجع و مرور تقسیم‌بندی می‌گردد.

۱-۱- هدف

هدف از ایجاد مستند، شرح داده می‌شود.

۱-۲- محدوده

در این قسمت، خلاصه‌ای از محدوده مستند بیان می‌گردد.

۱-۳- مراجع

در این قسمت، لیست مستندات ارجاع شده در این مستند، آورده می‌شود.

۱-۴- مرور

در این قسمت، بخش‌های سند و چگونگی بخش‌بندی آن‌ها، شرح داده می‌شود.

۲- جایگاه محصول^۱

۲-۱- شرح مسئله

در این بخش، مسئله‌ای که ساخت سیستم آن را حل خواهد کرد، بیان می‌گردد. روش تدوین این قسمت در فصل سوم آمده است.

۲-۲- شرح جایگاه محصول^۲

در همین فصل تشریح شده است.

۳- شرح کاربران و ذینفعان

در این جا، خلاصه‌ای از مشخصات کاربران و ذینفعان پژوهه ذکر می‌شود. چگونگی تدوین

1. Positioning

2. Product position statement

این قسمت و قسمت‌های زیرین آن، در فصل سوم بیان شده است.

۱-۳- ذینفعان

۲-۳- کاربران

۳-۳- محیط کاربر

۴-۳- نیازهای اصلی ذینفعان و کاربران
در همین فصل تشریح شده است.

۴- مرور محصول^۱

این قسمت و قسمت‌های زیرین آن، در همین فصل تشریح شده است.

۱-۴- دورنمای محصول^۲

۲-۴- مفروضات و وابستگی‌ها^۳

۵- ویژگی‌های محصول^۴

در همین فصل تشریح شده است.

۶- قیدها

روش تدوین قیدها در فصل سوم آمده است.

۷- ترتیب و اولویت‌بندی^۵

در همین فصل تشریح شده است.

۸- سایر نیازمندی‌های محصول^۶

در همین فصل تشریح شده است.

۹- نیازمندی‌های مستندسازی^۷

در همین فصل تشریح شده است.

۳-۸- اندازه سند چشم‌انداز

یکی از اهداف مستندسازی برقراری ارتباط بین افراد است. با توجه به هدف مذکور، رعایت موارد زیر در تدوین مستندات توصیه می‌گردد:

۱. خوانندگان مستند را در شروع نگارش هر مستند شناسایی کنید.

۲. مطمئن شوید آنچه را مستندسازی می‌کنید، خوانندگان به درستی خواهند فهمید.

به دلیل اهمیت سند چشم‌انداز، رعایت توصیه‌های قبلی در تدوین این سند باید با دقت بیشتری انجام گردد.

1. Product overview
2. Product perspective
3. Assumptions and dependencies
4. Product features
5. Precedence and priority
6. Other product requirements
7. Documentation requirements

صرف نظر از اندازه و محتوای سند چشم انداز، هدف از تهیه آن، کسب توافق ذینفعان در مورد سیستم است. برای دست یابی به این هدف، سند چشم انداز می‌تواند خلاصه چند صفحه‌ای از جلسه برگزار شده با ذینفعان باشد که در آن محدوده سیستم مورد توافق قرار گرفته است (برای پروژه‌های کوچک). چشم انداز می‌تواند مستندی حاوی چندین صفحه (برای پروژه‌های متوسط) باشد. همچنین این سند می‌تواند شامل چندین مستند جدا باشد، که هر یک، چشم انداز سیستمی از یک سیستم بزرگ هستند (پروژه‌های بزرگ).

۹- نکات کلیدی

- سند چشم انداز، یکی از مهم‌ترین مستندات پروژه‌های نرم‌افزاری است.
- سند چشم انداز، عامل تبادل نظرات و کسب توافق درباره «چرایی و چیستی» پروژه است.
- این سند، معیاری برای سنجش تصمیماتی است که در ادامه پروژه اتخاذ می‌گردند.

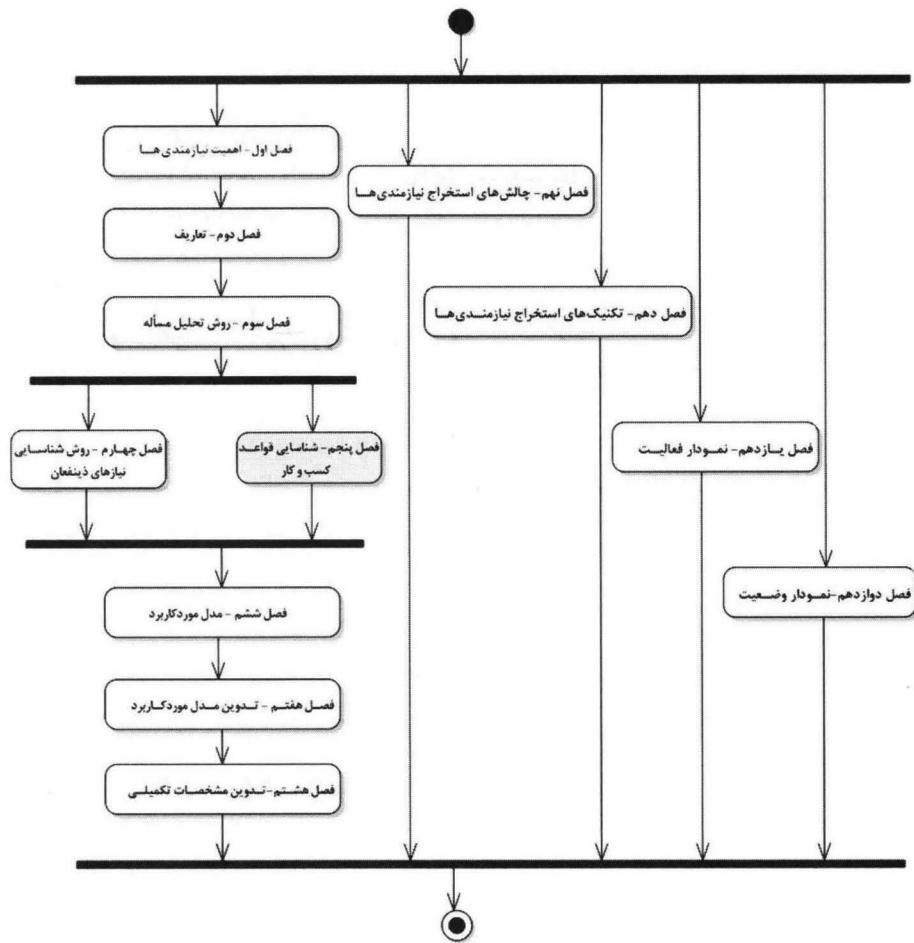
۱۰- مراجع

1. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
2. Leffingwell, Dean ,Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
3. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
4. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>

فصل پنجم - شناسایی قواعد کسب و کار

مهمترین عامل افزایش پیچیدگی، پذیرش اغلب ویژگی‌های مورد درخواست کاربران از سوی تولیدکنندگان نرم‌افزار است.

نیکولاوس ویرث طراح زبان پاسکال



۱ - مقدمه^۱

قواعد کسب و کار در توسعه سیستم های نرم افزاری دارای اهمیت و جایگاه ویژه ای هستند. در این فصل پس از تعریف، اهمیت، خاستگاه، سطح بندی، طبقه بندی قواعد کسب و کار و چگونگی شناسایی و تدوین آنها بیان می گردد.

۲ - تعریف قاعده کسب و کار

«قاعده کسب و کار» عبارتی کوتاه درباره کسب و کار و شرحی از یک سیاست^۲ یا شرط است که رعایت آن الزامی است. قواعد نوعی قید و توصیف گر مجموعه ای از شرط ها هستند. «قاعده کسب و کار» شرح فرآیند کسب و کار نیست، بلکه شرایطی است که فرایند در آن اجرا می گردد یا انتظار می رود پس از اتمام فرایند محقق شده باشند.

ممکن است قواعد، حاصل مجموعه ای از قوانین و آیین نامه های کسب و کار باشند. برای مثال پرداخت نقدی در شعب بانک ها تا سقف پانزده میلیون تومان در روز، قانونی است که بانک مرکزی (مرجع سیاست گذاری سیستم بانکی) ابلاغ نموده است و ضروری است نرم افزارهای بانکی آن را کنترل کنند.

هر چند توصیف کسب و کار نیاز به تعریف صدھا قاعده دارد اما هر قاعده به تنها ی قابل فهم، شناسایی، تعریف و بازبینی است.

قاعده کسب و کار به شکل های زیر تعریف می گردد:

- ﴿ «قاعده کسب و کار» بیانگر شرط یا قیدی است که در کسب و کار وجود دارد. ﴾
- ﴿ «قاعده کسب و کار» جزیی غیر قابل جداسازی از منطق کسب و کار است. ﴾
- ﴿ «قاعده کسب و کار» مفهومی است که با استفاده از واژه ^۳، واقعیت^۴ و قاعده^۵ توصیف می شود. ﴾
- نمونه هایی از قواعد مرتبط با سیستم حساب سپرده کوتاه مدت عبارتند از:
 - ﴿ حساب می تواند بیش از یک صاحب حساب داشته باشد. ﴾
 - ﴿ هر مشتری می تواند تنها یک حساب داشته باشد. ﴾
 - ﴿ برداشت مبلغ بیش از پانزده میلیون تومان پول نقد از یک حساب در روز مجاز نیست. ﴾

۱- در پژوهه های توسعه سیستم های اطلاعاتی، معمولاً فرایندهای کسب و کار فعلی، پیشنهادی یا هردو مستند و مدل می شوند که منجر به شناسایی قواعد کسب و کار نیز می گردد. این کار در گام «تحلیل مسئله» انجام می شود. از آنجا که در این کتاب، مدل سازی و مستند سازی کسب و کار معرفی نشده است، شناسایی قواعد کسب و کار به این بخش، منتقل شده است.

2. Policy

3. Term

4. Fact

5. Rule

- ﴿ سود به حسابی تعلق می‌گیرد که موجودی اش حداقل ۵۰،۰۰۰ تومان باشد.
- ﴿ مشتری خاص به مشتری‌ای گفته می‌شود که یکی از شرایط زیر را داشته باشد:
 - موجودی حساب مشتری بیش از یک صد میلیون تومان باشد.
 - میانگین موجودی حساب مشتری برای حداقل یک سال اخیر بیشتر از بیست میلیون تومان باشد.
- ﴿ کارمزد انتقال وجه تا مبلغ ۱۰ میلیون تومان از مشتریان خاص دریافت نمی‌گردد.

۳- اهمیت قواعد کسب و کار

سیستم اطلاعاتی برای تسریع و تسهیل کسب و کار تولید می‌گردد. نیل به هدف مذکور با رفع درست و مناسب نیازها در سیستم میسر است که لازمه‌اش شناخت کسب و کار است. شناسایی قواعد بخش پیاده‌سازی می‌گردد، اهمیت بسزایی دارد. لذا پس از شناسایی و مستندسازی، قواعد مذکور با کارفرما بازنگری می‌گردد تا تیم توسعه نرم‌افزار از شناخت درست قواعد مطمئن گردد.

قواعد، ورودی مهمی برای مرحله نیازمندی‌ها است. چرا که ضروری است در راه حل پیشنهادی وجود داشته باشند. به عنوان مثال گروه تحلیل نشان می‌دهد در مورد کاربرد «برداشت از حساب»، قاعده «برداشت از حساب مسدود مجاز نیست» کنترل شده است.

قواعد کسب و کار برای گروه طراحی و پیاده‌سازی نیز اهمیت بسزایی دارد. گروه یاد شده، قواعد را در قالب نیازمندی‌ها و مدل تحلیل تحويل می‌گیرد و راه حل‌هایی را برای اعمال و کنترل آن‌ها در سیستم تعییه می‌کند. به عنوان مثال قاعده «هر حساب تنها به یک مشتری تعلق دارد» در طراحی بانک اطلاعاتی با ایجاد کلید خارجی^۱ در جدول حساب به جدول مشتری اعمال می‌گردد. همچنین در صورت وجود قاعده «هر مشتری می‌تواند تنها یک حساب داشته باشد»، طراح بانک اطلاعاتی می‌تواند با تعریف کلید یکتا^۲ بر اساس کد مشتری در جدول حساب، قاعده را در سیستم اعمال کند.

کنترل درستی قواعد کسب و کار بخش مهمی از آزمون نرم‌افزار است. آزمونگر بر اساس قواعد، آزمایه‌هایی^۳ را اجرا می‌کند و از درستی پیاده‌سازی آن‌ها مطمئن می‌گردد. به عنوان مثال آزمون گر برای اطمینان از کنترل قاعده «هر مشتری می‌تواند تنها یک حساب داشته باشد» سعی می‌کند برای یک مشتری مشخص دو حساب افتتاح نماید. در صورت پیاده‌سازی درست قاعده، سیستم اجازه افتتاح دو میں حساب را نخواهد داد.

1. Foreign key
2. Unique key
3. Test cases

۴- خاستگاه عمومی قواعد کسب و کار

شناخت خاستگاه قواعد کسب و کار به شناسایی دقیق‌تر آن‌ها منجر می‌گردد. قواعد کسب و کار بخشی از روش‌های جاری یا آینده کسب و کار هستند، از این رو می‌توان ساختار، رفتار و مفاهیم کسب و کار را خاستگاه این قواعد دانست.

ساختار کسب و کار

ساختار کسب و کار به عنوان یکی از خاستگاه‌ها منجر به تعریف قواعد ساختاری می‌شود. قواعد یاد شده قیدها، روابط اجزا و مفاهیم کسب و کار را نشان می‌دهند. به عنوان مثال به قاعده زیر در حساب سپرده کوتاه مدت توجه کنید:

قاعده: هر مشتری می‌تواند چندین حساب سپرده کوتاه مدت داشته باشد.
عبارت قبلی بیانگر رابطه ساختاری بین حساب سپرده کوتاه مدت و مشتری است که از مفاهیم موجود در «شعبه» و «بانک» هستند.

رفتار کسب و کار

این خاستگاه منجر به شناسایی قواعد رفتاری می‌گردد که ترتیبی از کارها را تعریف می‌کند. این قواعد نحوه شناسایی و پاسخ‌گویی به رخدادهای کسب و کار را نیز مشخص می‌کنند. قاعده زیر را در حساب سپرده کوتاه مدت در نظر بگیرید:

قاعده: لازم است برداشت بیش از پنج میلیون تومان از حساب به تأیید رئیس شعبه رسانده شود.
این قاعده تغییر در عملکرد شعبه را در صورت درستی شرط «پرداخت بیش از پنج میلیون تومان» نشان می‌دهد (الزام تأیید رئیس شعبه).

مفاهیم کسب و کار

تعاریف مفاهیم کسب و کار، منجر به شناسایی قواعد تعریفی می‌گردند. قواعد حاصل، اصطلاح جدیدی را تعریف یا رابطه اصطلاحات موجود را بیان می‌کند. برای مثال در حساب سپرده کوتاه مدت قاعده زیر را در نظر بگیرید:

- ﴿ مشتری خاص به مشتری ای گفته می‌شود که یکی از شرایط زیر را داشته باشد:
 - موجودی حساب مشتری بیش از یکصد میلیون تومان باشد.
 - میانگین موجودی حساب مشتری برای حداقل یک سال اخیر بیشتر از بیست میلیون تومان باشد.

با تعریف مشتری خاص در قاعده قبلی، از آن در تعریف قاعده زیر استفاده می‌گردد.

﴿ کارمزد انتقال وجهه تا مبلغ ۱۰ میلیون تومان از مشتریان خاص دریافت نمی‌گردد.
انتخاب دسته‌بندی قواعد هدف نیست، بلکه شناسایی درست و کامل قواعد کسب‌وکار هدف است. دسته‌بندی نیل به هدف یاد شده را ساده‌تر می‌کند از این رو در انتخاب دسته‌بندی هر قاعده وسوسن نداشته باشد. ۲

۵- سطح‌بندی قواعد کسب‌وکار

قواعد کسب‌وکار بر اساس اهمیت و جزئیات به شکل زیر سطح‌بندی می‌گردد.

- ﴿ قواعد حاکم^۱
- ﴿ قواعد عملیاتی^۲
- ﴿ قواعد خودکار^۳

۱-۵- قواعد حاکم

این قواعد شامل احکام حقوقی (مانند قوانین، مصوبات و احکام قضایی)، مقررات و توافقات الزام‌آور (مانند تعهدات قراردادی) هستند. اغلب قواعد یادشده، با هدف رونق یا کنترل کسب‌وکار، تنظیم تعاملات با طرفین یا کاهش مخاطرات و تهدیدات اعمال می‌شوند. این گونه قواعد به مجموعه‌ای از قواعد عملیاتی تفسیر می‌شوند تا در فرایندهای کسب‌وکار یا طراحی سیستم‌های نرم‌افزاری قابل استفاده باشند.

به عنوان مثال، قاعده زیر را که بخش‌نامه بانک مرکزی است در نظر بگیرید:

قاعده: دریافت مبلغی بیش از پانزده میلیون تومان پول نقد از حساب در روز مجاز نیست.

نحوه عملیاتی شدن بخش نامه را در دو بانک عامل با شرایط زیر در نظر بگیرید:

حال اول: در بانک عامل، به دلیل نبود سیستم نرم‌افزاری متتمرکز، برداشت از حساب تنها در شعبه افتتاح کننده مقدور است.

حال دوم: بانک عامل دارای سیستم متتمرکزی است که شعب به کمک آن وظایفشان را انجام می‌دهند.

در حالت اول، بانک عامل طی دستورالعملی، بخش نامه را به اطلاع شعب می‌رساند. از آن پس رئیس شعبه مراقب خواهد بود تا دستورالعمل به درستی اجرا گردد. بازرسان اجرای دستورالعمل، شعب را کنترل خواهند کرد. در حالت دوم، ابتدا قاعده در سیستم نرم‌افزاری شعبه پیاده‌سازی می‌گردد، سپس به همراه ارسال دستورالعمل به کلیه شعب، نسخه جدید سیستم نیز راه اندازی می‌شود و مورد بهره برداری قرار می‌گیرد.

1. Governing rules
2. Operational rules
3. Automated rules

روش اجرای بخش نامه در دو بانک عامل به یک شیوه نبوده است.

۲-۵- قواعد عملیاتی

این دسته از قواعد بیانگر قواعد عملیات کسب و کارند. این گونه قواعد برای طراحی و تولید سیستم های نرم افزاری مناسب است. مثال زیر قاعده‌ای عملیاتی است که در سیستم حساب سپرده کوتاه مدت پیاده‌سازی خواهد شد:

قاعده: موجودی قابل برداشت مساوی است با موجودی حساب منهاهی حداقل موجودی حساب منهاهی جمع مبالغ مسدود شده.

۳-۵- قواعد خودکار

قاعده خودکار، قاعده‌ای است که مطابق فناوری منطق کسب و کار^۱ یا نرم افزار بیان می‌شود. این گونه قواعد، پیاده‌سازی شده قواعد عملیاتی اند. به مثال زیر توجه کنید.

قاعده: نمی‌توان مشتریانی را که دارای حساب هستند از سیستم حذف کرد.
در کسب و کار عمل حذف مشتری بی معنی است. این مفهوم برخاسته از قابلیت حذف اطلاعات مشتری در سیستم نرم افزاری است.

۶- طبقه‌بندی قواعد کسب و کار

قواعد کسب و کار را می‌توان به صورت زیر دسته‌بندی کرد. مانند تمامی دسته‌بندی‌ها، هدف از این کار شناخت بهتر و شناسایی آسان تر قواعد کسب و کار است:

۱-۶- قواعد قیدی^۲

قواعد قیدی، سیاست‌ها یا شرایطی هستند که ساختار یا رفتار اشیاء را مقید می‌کنند. این گونه قواعد ممکن است تنها در شرایط خاصی معتبر باشند. به قیدهایی که همیشه معتبرند، قواعد ثابت^۳ گفته می‌شود.

قواعد قیدی به سه دسته قواعد محرك و پاسخ^۴، قواعد مقید‌کننده عملیات^۵ و قواعد مقید‌کننده ساختار^۶ تقسیم می‌گردند.

-
1. business logic technologies
 2. Constraint rules
 3. invariants
 4. Stimulus and response rules
 5. Operation constraint rules
 6. Structure constraint rules

- قواعد محرک و پاسخ: قواعد محرک و پاسخ، انجام رفتاری را به زمان و شرایط مشخصی مقید می‌کنند. در مثال زیر پرداخت سود (رفتار مورد نظر) به گذشت حداقل یک ماه از افتتاح حساب (شرایط مشخص) مقید شده است.

قاعده: هنگام بستن حساب، سود در صورتی تعلق می‌گیرد که یک ماه از افتتاح آن سپری شده باشد.

- قواعد مقید کننده عملیات: قواعد مقید کننده عملیات، شرایطی را مشخص می‌کنند که برای اطمینان از درستی انجام عمل قبل و یا بعد از آن، باید برقرار باشند. در مثال زیر شرط برداشت از حساب (عمل) مسدود نبودن حساب تعیین شده است.

قاعده: برداشت از حساب مسدود، مجاز نیست.

- قواعد مقید کننده ساختار: قواعد مقید کننده ساختار، قواعدی هستند که سیاست‌ها یا شرایطی را درباره کلاس‌ها، اشیاء و روابط آن‌ها بیان می‌کند.

قاعده: هر حساب بانکی کوتاه مدت تنها به یک مشتری تعلق دارد.

۲-۶- قواعد اشتقة^۱

این گونه قواعد، سیاست‌ها یا شرایطی را برای استنتاج یا محاسبه واقعیت^۲ از روی سایر واقعیت‌ها تعریف می‌کنند.

- این قواعد به دو دسته قواعد استنتاجی و قواعد محاسباتی تقسیم می‌گردند.
- قواعد استنتاجی: این‌گونه قواعد، بیان کننده نتیجه‌ای هستند که از استنتاج منطقی مجموعه‌ای از واقعیت‌ها به دست می‌آیند.

قاعده: حساب غیرفعال به حسابی گفته می‌شود که به مدت شش ماه، برداشت از یا واریزی به آن انجام نشده باشد.

 - قواعد محاسباتی^۳: این گونه قواعد بیان کننده نتایجی هستند که به کمک فرمول‌های ریاضی توصیف می‌گردند. قواعد یاد شده، حالت پیچیده‌تر قواعد استنتاجی هستند.

قاعده: سود حساب سپرده کوتاه مدت در هر روز به شکل زیر محاسبه می‌شود.

حداقل موجودی حساب در روز* سود سالانه حساب سپرده کوتاه مدت = سود روزانه

۷- شناسایی قواعد کسب و کار

قواعد کسب و کار در روال های جاری، قوانین، مستندات یا ذهن خبرگان کسب و کار نهفته است. استخراج قواعد کسب و کار مشابه استخراج نیازمندی ها در توسعه نرم افزار است. از این رو روش های ارائه شده برای استخراج نیازمندی ها می تواند در شناسایی قواعد کسب و کار نیز به کار گرفته شود. روش های یاد شده در فصل ده معرفی شده اند.

۸- تدوین قواعد کسب و کار

قواعد کسب و کار به صورت مدل و سند تدوین می گردند. تعریف قواعد کسب و کاری که بیانگر قیدهای ساختاری و رفتاری است، به شکل مدل ساده‌تر است. بهتر است سایر قواعد به خصوص محاسبات و سیاستها مستند گردند. قواعد کسب و کار می توانند در سایر مستندات مانند واژه‌نامه، مشخصات مورد کاربرد کسب و کار^۱، سند چشم‌انداز و سند مشخصات مورد کاربرد نیز آورده شوند. در ادامه برخی از شیوه های تدوین قواعد کسب و کار تشریح می گردد.

۸-۱- تدوین قواعد در مدل (مدل سازی)^۲

قواعد کسب و کار را بر حسب نوع و ماهیت آنها می توان مدل سازی کرد. به طور معمول نمودارهای زیر در مدل سازی استفاده می شود:

- ﴿ نمودار کلاس^۳ ﴾
- ﴿ نمودار فعالیت ﴾
- ﴿ نمودار وضعیت ﴾

۸-۱-۱- نمودار کلاس

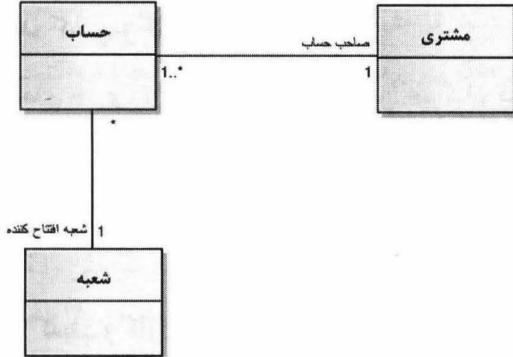
اجزای نمایشی نمودار کلاس می توانند بیان گر قواعد کسب و کار باشند. به عنوان مثال در نمودار زیر که بیان گر رابطه مشتری، حساب و شعبه در سیستم حساب سپرده کوتاه مدت است، قواعد زیر نهفته است:

- ﴿ حساب به یک مشتری تعلق دارد. (یعنی حساب مشترک- حسابی که دارای چندین صاحب حساب باشد- وجود ندارد)
- ﴿ هر مشتری می تواند بیش از یک حساب داشته باشد.
- ﴿ مشتری کسی است که حداقل یک حساب داشته باشد.

1. Business use case specification

2- بعضی از روش ها به جای مدل و مستندسازی قواعد کسب و کار، تنها مدل سازی را پیشنهاد می کنند.

3. Class diagram

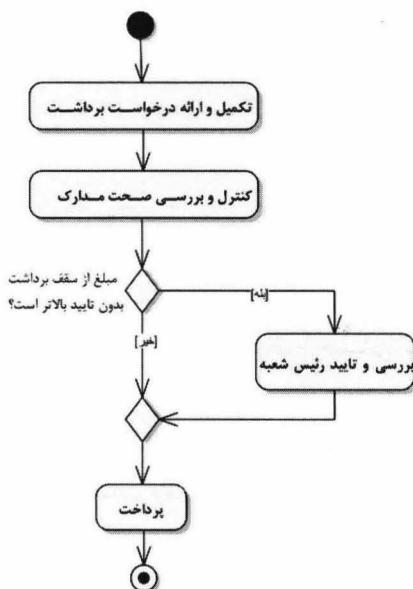


شکل ۵-۵: مدل‌سازی قواعد کسب‌وکار در نمودار کلاس

۲-۱-۸- نمودار فعالیت

نمودار فعالیت نیز یکی از گزینه‌هایی است که می‌تواند برای مدل‌سازی قواعد کسب‌وکار به کار رود. شکل زیر نشان می‌دهد که مشتری پس از تکمیل درخواست برداشت، آن را به متصدی امور بانکی تحویل می‌دهد. وی مدارک مشتری و درستی فرم درخواست را برسی می‌کند. در صورتی که مبلغ برداشت از «سقف برداشت بدون تأیید» بیشتر باشد، تأیید رئیس شعبه برای پرداخت به مشتری الزامی است. پس در این نمودار قاعده زیر مدل‌سازی شده است:

﴿ برای برداشت مبالغ بیش از «سقف برداشت بدون تأیید» از حساب، تأیید رئیس شعبه الزامی است. ﴾

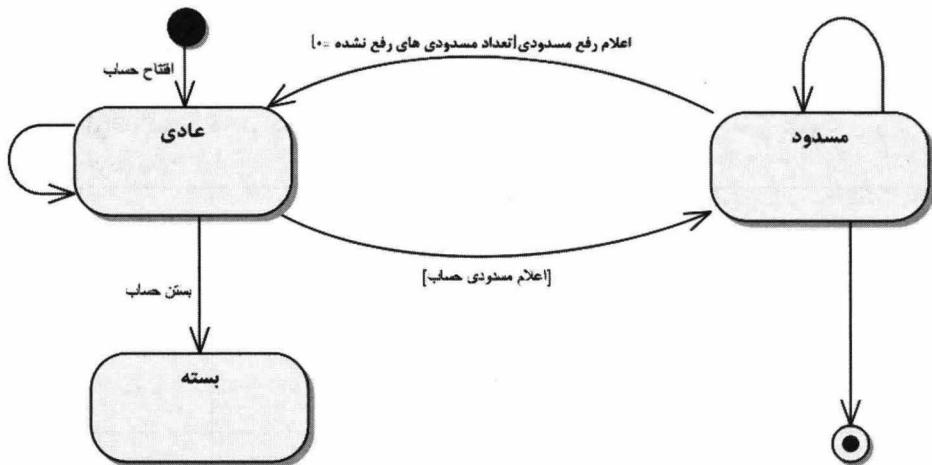


شکل ۵-۶: مدل‌سازی قواعد کسب‌وکار در نمودار فعالیت

۳-۱-۸- نمودار وضعیت

نمودار وضعیت نیز از نمودارهایی است که قواعد کسب و کار می‌توانند در آن مدل شوند. در شکل زیر وضعیت‌های حساب نشان داده شده است. حساب در طول فعالیتش می‌تواند وضعیت‌های «عادی»، «مسدود» و «بسته» را داشته باشد. با افتتاح یک حساب، آن حساب در وضعیت «عادی» قرار می‌گیرد. در صورتی که حساب بسته شود، وضعیت آن به «بسته» تغییر می‌کند. هنگامی که دستور بانک یا حکم مرجع قضایی برای مسدود کردن حساب در شعبه دریافت گردد، حساب «مسدود» می‌شود. پس در نمودار وضعیت، قواعد کسب و کار زیر مدل شده‌اند:

- « حساب با دستور مرجع قضایی یا بانک مسدود می‌گردد.
- « بستن، واریز و برداشت از حساب مسدودی مجاز نیست.
- « تا رفع تمام مسدودی‌های قبلی، حساب مسدود باقی خواهد ماند.



شکل ۳-۵: مدل‌سازی قواعد کسب و کار در نمودار وضعیت

۲-۸- تدوین قواعد در سند قواعد کسب و کار

مستند قواعد کسب و کار^۱، مستندی برای تدوین و مستندسازی قواعد کسب و کار است. قالب^۲ این سند به شکل زیر است.

۱- مقدمه

در این قسمت از مستند مقدمه‌ای بر مستند نوشته می‌شود. مقدمه در قالب چهار بخش هدف، محدوده، مراجع و مرور تقسیم‌بندی می‌گردد.

۱-۱- هدف

شرحی از هدف ایجاد مستند بیان می‌شود.

۱-۲- محدوده

شرح مختصری درباره محدوده این مستند در این بخش نوشته می‌شود.

۱-۳- مراجع

در این بخش لیست کاملی از استنادی که در این مستند به آن‌ها ارجاع داده شده یا از آن‌ها استفاده شده ارائه می‌گردد.

۱-۴- مرور

در این بخش چگونگی سازماندهی و بخش‌های مستند تشریح می‌گردد.

۲- شرح قواعد کسب‌وکار

محتوای اصلی سندهای در این بخش ارائه می‌شود.

۲-۱- [دسته^۱] قواعد کسب‌وکار^۲

دسته‌بندی قواعد کسب‌وکار موجب خوانایی بیشتر می‌شود. می‌توان آن‌ها را در گروه‌هایی دسته‌بندی و در این بخش ذکر کرد.

ردیف	کد قاعدة	[عنوان قاعدة] شرح قاعدة

شرح هر یک از ستون‌ها در زیر آمده است.

» ردیف

» کد قاعدة: کد یکتاًی که به هر قاعدة داده می‌شود.

» عنوان قاعدة: عنوان قاعدة کسب‌وکار

» شرح قاعدة: مشرح قاعدة کسب‌وکار

جدول زیر بخشی از مستند قواعد کسب‌وکار سیستم حساب سپرده کوتاه مدت است.

1. Category

۲- ممکن است، دسته‌بندی خاصی تعیین نگردد. در این صورت این بخش وجود نخواهد داشت.

جدول ۵-۱: نمونه مستند قواعد کسب و کار

ردیف	کد قاعدہ	[عنوان قاعدہ] شرح قاعدہ
۱	BR001	مبلغ قابل برداشت: مبلغ قابل برداشت برابر است با موجودی حساب منهای مجموع مبلغ های مسدودی حساب اعلام کد ملی واریز کننده در فرم درخواست واریز اجباری است.
۲	BR002	

در فصل هفتم چگونگی ارتباط بین قواعد کسب و کار و نیازمندی ها به خصوص مورد کاربردها توضیح داده شده است.

۳-۸- تدوین قواعد در سایر اسناد

قواعد کسب و کار را می توان در سایر مستندات برای توسعه سیستم نیز تدوین نمود. واژه نامه و مشخصات مورد کاربرد دو نمونه از این مستندات هستند:

﴿ واژه نامه ^۱ ﴾

واژگان در این مستند تعریف می گردد. مستند مذکور محل مناسبی برای تدوین قواعد با خاستگاه «تعاریف» است. برای نمونه، تعریف مشتری خاص و حساب را کد که توصیف گر قواعدی هستند می تواند در سند واژه نامه ذکر شود.

مشتری خاص:

- مشتری خاص به مشتری ای گفته می شود که یکی از شرایط زیر را دارا باشد:
- موجودی حساب مشتری بیش از یک صد میلیون تومان باشد.
- میانگین موجودی حساب مشتری برای بیش از یک سال بیشتر از بیست میلیون تومان باشد.

﴿ سند مشخصات مورد کاربرد ﴾

سند مشخصات مورد کاربرد ^۲ تعاملات بین سیستم و کاربر را بیان می کند. مستند یادشده می تواند حاوی قواعدی از کسب و کار باشد که در زمان انجام تعامل رعایت می شوند. به عنوان مثال در مورد کاربرد «افتتاح حساب» چنان چه مشتری قبل حسابی در بانک داشته باشد، سیستم از افتتاح حساب جلوگیری خواهد کرد. این کار به دلیل وجود قاعدة «مشتری می تواند تنها یک حساب داشته باشد» انجام می شود.

گاهی به قالب مشخصات مورد کاربرد بخشی به نام «قواعد کسب و کار» اضافه می شود. این روش توصیه نمی گردد، چرا که روش یاد شده باعث تکرار شدن قواعد مشترک در چندین مستند خواهد گردید و نگهداری و تغییر آنها دشوار می گردد.

۱- در RUP دو مستند جداگانه، یکی برای واژه های کسب و کار و یکی برای واژه های سیستم وجود دارد. در این کتاب دو مستند تلفیق شده اند و واژه نامه نامیده شده است.

۲- use case specification در فصل ششم تشریح شده است.

۹- راهنمای تدوین قواعد کسب و کار

- در این بخش مجموعه‌ای از توصیه‌ها و راهنمایی‌ها برای تدوین قواعد کسب و کار ارائه شده است. هدف از ارائه آن‌ها اشاره به اشتباهات رایج در تدوین قواعد و کمک به تدوین درست آن‌ها است.
- توصیه ۵-۱: قاعده کسب و کار باید دقیق باشد.
 - نادرست: اگر موجودی کافی نباشد، سود به حساب تعلق نمی‌گیرد.
 - درست: اگر موجودی حساب مشتری کمتر از ۵۰،۰۰۰ تومان باشد، سود به حساب تعلق نمی‌گیرد.
 - توصیه ۵-۲: از بیان قواعد واضح و مشخص اجتناب گردد. این گونه عبارت‌ها را «گزاره‌های همیشه درست» می‌نامیم.
 - نادرست: حساب همواره دارای موجودی است.
 - توصیه ۵-۳: قواعدی که هرگز رخ نمی‌دهند، بیان نگردنند. این عبارات را «گزاره‌های همیشه نادرست» می‌نامیم.
 - توصیه ۵-۴: از بیان کلمات زاید پرهیز شود.
 - نادرست: به شعبه‌ای که مشتری به آن جا مراجعه می‌کند و پس از طی تشریفات و ارائه مدارک، حسابی افتتاح می‌کند، شعبه افتتاح‌کننده گفته می‌شود.
 - درست: به شعبه‌ای که حساب در آن افتتاح می‌گردد، شعبه افتتاح‌کننده گویند.
 - توصیه ۵-۵: در قاعده کسب و کار، مجری، چگونگی، مکان و زمان انجام قواعد آورده نشود.
 - نادرست: متصدی امور بانکی باید هنگام صدور سند دریافت یا پرداخت کنترل کند که سند، متوزن(بالانس) باشد.
 - درست: سند دریافت یا پرداخت باید متوزن(بالانس) باشد.
 - توصیه ۵-۶: قواعد محاسباتی تاحد امکان جداگانه مدون گردند.
 - مثال: برداشت از حساب به مبلغی بیشتر از موجودی حساب منهای مجموع مبلغ‌های مسدودی حساب مجاز نیست.
 - پیشنهاد:
 - قاعده ۱: مبلغ قابل برداشت برابر است با موجودی منهای مجموع مبلغ‌های مسدودی حساب
 - قاعده ۲: برداشت از حساب به مبلغی بیشتر از «مبلغ قابل برداشت» مجاز نیست.
 - توصیه ۵-۷: قاعده کسب و کار باید برای خبرگان حوزه مسئله^۱ قابل فهم باشد. یکی از مهم‌ترین دلایل وجود ابهام در قواعد کسب و کار، بیان آن‌ها با ادبیات تکنولوژی و نرم‌افزار است.

- نادرست: کد ملی در فرم درخواست واریز نمی‌تواند خالی باشد.
- درست: اعلام کد ملی واریزکننده در فرم درخواست واریز اجباری است.
- توصیه ۸-۵: نهاد یا فاعل به صورت مفرد بیان گردد.
- نادرست: مشتریان می‌توانند تا حداقل موجودی از حساب‌ها برداشت کنند.
- درست: مشتری می‌تواند تا حداقل موجودی از حساب برداشت کند.
- توصیه ۹-۵: قاعده کسب و کار باید «مشخص» باشد. از بکار بردن کلماتی مانند «و غیره»، «...» و «امثالهم» که احتمال ایجاد تفسیرهای مختلف و ابهام را تشیدید می‌کند، پرهیز کنید.
- نادرست: مدارک لازم برای افتتاح حساب عبارتند از شناسنامه، تصویر صفحه اول، کارت ملی و تصویر کارت ملی.
- توصیه ۱۰-۵: قواعد کسب و کار تا حد امکان به صورت جملات غیرشرطی بیان شوند.
- مثال: اگر حسابی مسدود شود نمی‌توان مبلغی از آن برداشت یا به آن واریز کرد.
- پیشنهاد: برداشت از حساب مسدود و واریز به آن مجاز نیست.
- توصیه ۱۱-۵: در نگارش قواعد کسب و کار از بیان مفاهیم سیستم‌های نرم‌افزاری اجتناب گردد. از جمله این مفاهیم می‌توان به ایجاد، مشاهده، ویرایش و حذف^۱ موجودیت‌ها^۲ اشاره کرد.
- نادرست: پس از ثبت اعلام مسدودی حساب، مبلغ قابل برداشت باید به روز شود.
- درست: مبلغ قابل برداشت برابر است با موجودی حساب منهای مجموع مبلغ‌های مسدودی حساب.
- توصیه ۱۲-۵: اگر قاعده قابل اعمال بر چند فاعل/نهاد باشد، از جمله بدون فاعل/نهاد استفاده شود.
- مثال: در صورتی که حساب مشتری فعال باشد، مشتری می‌تواند از حساب برداشت کند.
- پیشنهاد: در صورتی که حساب مشتری فعال باشد، می‌توان از حساب برداشت کرد.
- توصیه ۱۳-۵: از نوشتن جملات امری در بیان قواعد کسب و کار پرهیز شود.
- نادرست: در صورتی که مبلغ مورد نظر مشتری برای افتتاح حساب کمتر از ۵۰,۰۰۰ تومان باشد حساب را افتتاح نکنید.
- درست: حداقل مبلغ لازم برای افتتاح حساب ۵۰,۰۰۰ تومان است.
- توصیه ۱۴-۵: واژگانی که قواعدی را توصیف می‌نمایند یا در توصیف قاعده‌ای به کار می‌روند، جداگانه تعریف شوند.

1. CRUD (Create, Read, Update, Delete)

2. Entity

- مثال: مشتری که موجودی حساب وی بیش از بیست میلیون تومان در مدت شش ماه باشد، می‌تواند با تأیید رئیس شعبه، بیش از سه بار در روز از حساب برداشت کند.
- پیشنهاد:
 - قاعده ۱: مشتری خاص به مشتری‌ای گفته می‌شود که موجودی حساب وی برای شش ماه بیش از بیست میلیون تومان باشد.
 - قاعده ۲: سقف تعداد برداشت عبارت از حداقل تعداد دفعاتی که می‌توان در یک روز از حساب برداشت نمود.
 - قاعده ۳: مشتری خاص می‌تواند با تأیید رئیس شعبه، بیش از سقف تعداد برداشت از حساب برداشت کند.
- توصیه ۵-۱۵: جدول تصمیم، تکنیکی برای تدوین قواعد کسب‌وکار است که می‌توان آن را در شرایط زیر استفاده نمود:
 ۱. وقتی مقادیر قاعده را نتوان به صورت فرمول بیان نمود. به نمونه جدول ۵-۲ توجه کنید.

جدول ۵-۲: سود بانکی انواع حساب

سال ۸۱		سال ۸۰		نوع حساب
زودتر از سررسید	عادی	زودتر از سررسید	عادی	
...	...	-	۱۴	کوتاه مدت
...	...	۱۴	۱۶	بلند مدت یکساله
...	بلند مدت پنج ساله

۲. مجموعه‌ای از قواعد که دارای هدف مشترکی باشند.
 - سود سپرده کوتاه مدت در سال ۱۳۸۶، سالانه ۱۴ درصد است.
 - سود سپرده کوتاه مدت در سال ۱۳۸۷، سالانه ۱۲ درصد است
 - ...

جدول ۵-۳: سود حساب سپرده کوتاه مدت

درصد	سال
۱۷	۱۳۸۶
۱۲	۱۳۸۷

۳. وقتی که مفهوم به کارگرفته شده در قاعده دارای مقادیر محدودی باشد.
- توصیه ۵-۱۶: در تشریح قواعد کسب و کار به وضعیت های شیء توجه شود. در این شرایط به این نکات دقت شود:

- تعریف هر یک از وضعیت های شیء می تواند منجر به شناسایی قواعد جدیدی گردد.
- ممکن است هر وضعیت، مجموعه ای از قواعد خاص خود را داشته باشد.
- ممکن است تغییر شیء از وضعیتی به وضعیت دیگر، با مجموعه ای از قواعد همراه باشد.
- مثال ۱: حساب دارای وضعیت های عادی، مسدود و بسته است.
- مثال ۲: برداشت از حساب مسدود مجاز نیست.
- مثال ۳: وضعیت حساب مسدود هنگامی عادی خواهد شد که تمام اعلامیه های مسدودی آن، رفع مسدودی گردند.

- توصیه ۵-۱۷: در تشریح قواعد کسب و کار، به بازه های زمانی توجه شود. در این شرایط به این نکات دقت شود:
- تعریف بازه ها می تواند منجر به شناسایی قواعد جدیدی گردد.
 - ممکن است مجموعه ای از قواعد تنها در بازه های خاصی معتبر باشند.
 - معمولاً ورود از یک بازه به بازه دیگر منجر به تعریف مجموعه ای از قواعد می گردد.

- مثال ۱: روز کاری^۱ شعبه، بازه ای است بین «آغاز روز شعبه» و «پایان روز شعبه» که متصدیان امور بانکی در شعبه، امکان انجام عملیات بانکی را دارند.
- مثال ۲: کلیه عملیات شعبه تنها در یک روز کاری قابل انجام است.
- مثال ۳: در پایان روز کاری باید اسناد حسابداری قطعی گردد.

۱۰- نکات کلیدی

- » قاعده کسب و کار بیانگر شرط یا قیدی است که در کسب و کار وجود دارد.
- » پوشش کامل و صحیح قواعد کسب و کار در سیستم های نرم افزاری از عوامل موفقیت پژوهه است.
- » ساختار، رفتار و مفاهیم کسب و کار خاستگاه های اصلی قواعد کسب و کار هستند. برای شناسایی قواعد کسب و کار از روش های شناسایی نیازمندی ها استفاده می شود.

۱۱- مراجع

1. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
2. Leffingwell, Dean ,Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
3. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
4. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>



بخش چهارم: تعریف سیستم

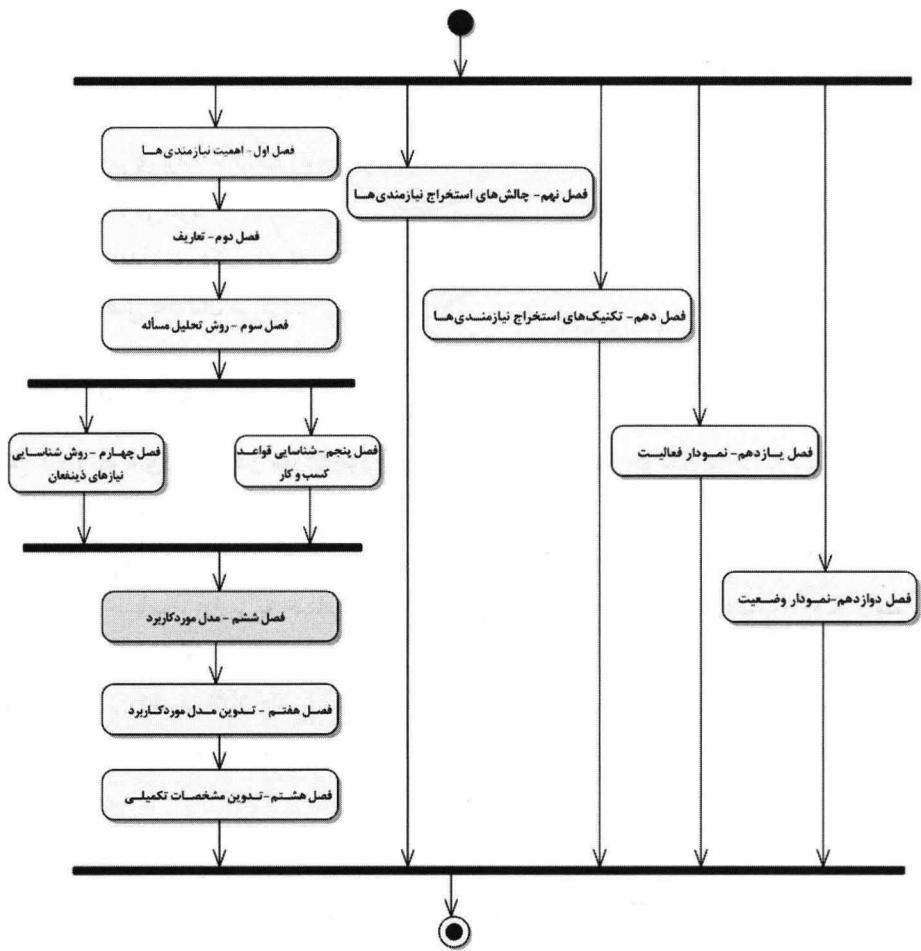
هدف این بخش، تعریف نیازمندی‌های تفصیلی سیستم بر اساس نیازمندی‌های کلان آن است که در مرحله قبلی تأیید شده است. این کار شامل تدوین موارد کاربرد، شناسایی نیازمندی‌های غیرکارکردی و تصحیح سند چشم‌انداز است.

فصل ششم – مدل موردکاربرد
فصل هفتم – تدوین مدل موردکاربرد
فصل هشتم – تدوین مشخصات تكمیلی

فصل ششم - مدل موردنکاربرد

راه رفتن روی آب و توسعه نرم افزار از روی مشخصات (*Specification*) آسان است اگر هر دوی آنها منجذب (غیرقابل تغییر) باشند.

ادوارد وی بی ارد



۱- مقدمه

پس از تدوین نیازهای ذینفعان و کلیات راه حل در قالب سند چشم‌انداز (فصل چهارم) و شناسایی قواعد کسب و کار به عنوان بخشی از صورت مسئله (فصل پنجم) لازم است راه حل پیشنهادی با جزئیات بیشتری ارائه گردد.

در این کتاب برای ارائه جزئیات راه حل از تکنیک «موردکاربرد^۱» استفاده شده است.^۲ نتیجه کار، «مدل موردکاربرد» نامیده می‌شود.

در این فصل «مدل موردکاربرد»، اهمیت، کاربرد و اجزای آن تشریح می‌گردد. روش تدوین «مدل موردکاربرد» در فصل بعد بیان می‌گردد.

۲- مدل مورد کاربرد

در شکل ۶-۱، نمونه‌ای از «نمودار موردکاربرد^۳» ارائه شده است که بخشی از مدل موردکاربرد در سیستم حساب سپرده کوتاه مدت است.

این تصویر نشان می‌دهد که متصدی امور بانکی -که آن را کنشگر^۴ می‌نامیم- می‌تواند کارهای «تعریف مشتری»، «افتتاح حساب»، «واریز به حساب»، «برداشت از حساب» و «بستن حساب» را -که آنها را موردکاربرد می‌نامیم- انجام دهد.

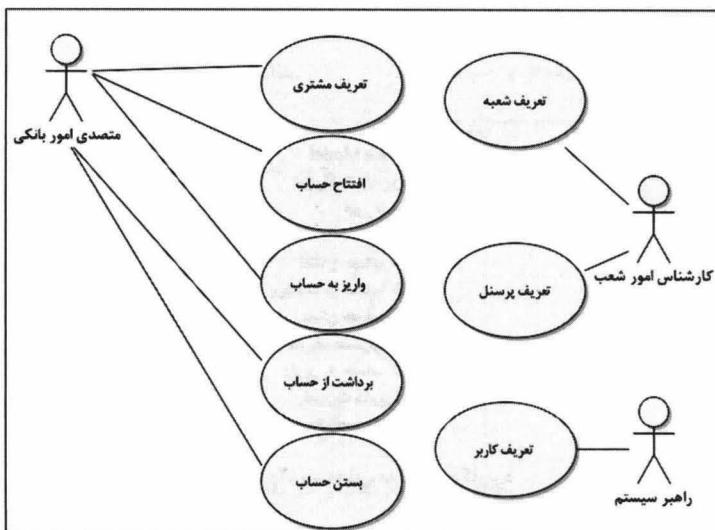
هم‌چنین در این تصویر نشان داده شده است که راهبر سیستم می‌تواند «تعریف کاربر» را انجام دهد و کارشناس امور شعب نیز «تعریف شعبه» و «تعریف پرسنل» را انجام می‌دهد.

1. Use case

۲- سایر تکنیک‌ها در فصل ۱۰ معرفی شده‌اند.

3. Use case diagram

4. Actor



شکل ۶-۱: یک نمودار مورد کاربرد در سیستم حساب سپرده کوتاه مدت

تعامل بین سیستم و کنشگر (متصرف امور بانکی) برای انجام «تعریف مشتری»^۱ به شکل زیر است:

- کاربر گزینه «تعریف مشتری» را انتخاب می‌کند.
- سیستم فرم «تعریف مشتری» را نمایش می‌دهد.
- کاربر اطلاعات مشتری را به شرح ذیل وارد می‌کند:
 - * شماره مشتری^۲ ●
 - * نام^۳ ●
 - * نام خانوادگی^{*} ●
 - * کد ملی^{*} ●
- کاربر گزینه «تایید» را انتخاب می‌کند.
- سیستم اطلاعات مشتری را ذخیره می‌کند.

گام‌های مذکور بخشی از تعامل طرفین برای نیل به هدفی مشخص - در اینجا تعریف مشتری - است.
گام‌های بالا در مستند «مشخصات مورد کاربرد»^۴ «تعریف مشتری» مدون می‌گردند. دو مورد کاربرد دیگر نیز
دارای مستند «مشخصات مورد کاربرد» مختص به خود هستند.

در شکل زیر نمایی از مدل مورد کاربرد در ابزار EA^۵ نشان داده شده است که موارد کاربرد در قالب

- ۱- برای سادگی فرض شده است که اطلاعات مشتری تنها شامل موارد ذکر شده است. در بخش‌های بعدی این مشخصات به طور کامل ارائه خواهد شد.
- ۲- در اینجا فرض شده است که کاربر شماره مشتری را وارد می‌کند و سیستم آن را تولید خواهد کرد.
- ۳- علامت * به معنای اجباری بودن داده‌های ورودی است.

بسته‌های^۱ «شعبه»، «امور شعب» و «مدیریت کاربران» دسته‌بندی شده‌اند. بسته‌های مذکور، مدل مورد کاربرد را به اجزای منطقی کوچک‌تری تقسیم کرده‌اند.



شکل ۶-۲: بسته‌های مدل مورد کاربرد

مدل مورد کاربرد، مدلی از تعامل کاربر با سیستم برای تحقق اهداف مورد انتظار است. به عبارت دیگر، مدل مورد کاربرد، مدلی از نیازمندی‌های کارکردی سیستم است که مبنی بر موارد کاربرد بیان می‌گردد. این مدل شامل اجزای ذیل است:

- کنشگر
- مورد کاربرد
- نمودار مورد کاربرد
- مشخصات مورد کاربرد
- بسته
- سند مرور مدل مورد کاربرد^۲

در ادامه فصل، هر یک از اجزای مدل مورد کاربرد تشریح می‌گردد.

۳- کنشگر

در این بخش، کنشگر به عنوان یکی از اجزای مدل مورد کاربرد معرفی می‌گردد.

۱-۳- تعریف کنشگر

برای مدل سازی راه حل پیشنهادی، ابتدا باید مشخص نمود که «سیستم برای چه کسانی طراحی شده است» و «با چه سیستم‌ها و تجهیزات سخت‌افزاری^۳ در ارتباط است». برای این کار، محیط سیستم به دو بخش تقسیم می‌گردد:

1. Packages
2. Use case model survey
3. Hardwares devices

○ سیستم (درون سیستم)

○ اجزای خارجی در ارتباط با سیستم

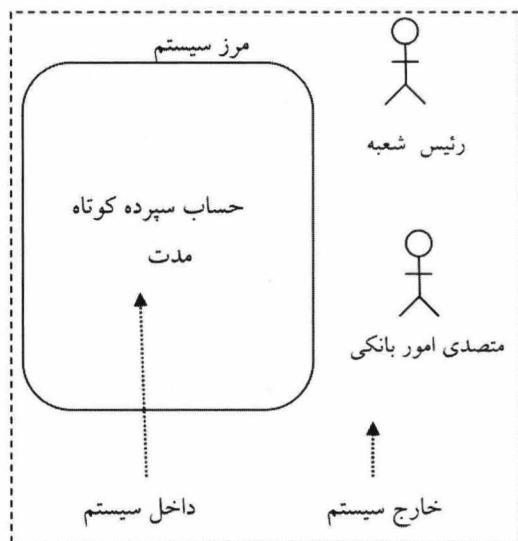
اجزای خارجی مرتبط با سیستم کنشگر نامیده می‌شوند و به شکل زیر تعریف می‌گردند:
 «کنشگر، کسی یا چیزی خارج از سیستم است که با آن در تعامل است».
 در زبان UML کنشگر به شکل زیر نمایش داده می‌شود.



نام کنشگر

شکل ۶-۳: نماد کنشگر در UML

در سیستم حساب سپرده کوتاه مدت، متصلی امور بانکی، رئیس شعبه، کارشناس امور شعب و سیستم اطلاعات مشتریان، کنشگرهای سیستم محسوب می‌گردند.



شکل ۶-۴: کنشگرهای سیستم حساب کوتاه مدت

۲-۳ - انواع کنشگر

همان طور که قبلاً در فصل دوم نیز ذکر شد یکی از اهداف طبقه‌بندی در علوم، شناخت بهتر پدیده‌ها و تمکن بر ویژگی‌های خاص هر طبقه است. به همین منظور کنشگرهای زوایای مختلفی قابل طبقه‌بندی هستند که در ادامه به برخی از آن‌ها اشاره شده است.

۱-۲-۳- طبقه‌بندی بر اساس ماهیت فیزیکی

کنشگرها بر اساس ماهیت به سه طبقه زیر تقسیم می‌گردند:

- انسان‌ها: کاربرانی که با سیستم کار می‌کنند.
- سیستم‌های نرم‌افزاری: سایر سیستم‌های نرم‌افزاری که با سیستم در تعامل هستند.
- تجهیزات سخت‌افزاری: دستگاه‌های سخت‌افزاری مانند پویش‌گر^۱، رمزینه‌خوان^۲ و کارت‌خوان که در سیستم مورد استفاده قرار می‌گیرند.

سیستم برای ارتباط با دنیای خارج، از رابط^۳ استفاده می‌کند. برای ارتباط با کاربران از رابط کاربر^۴، برای سیستم‌های نرم‌افزاری از رابط‌های برنامه‌نویسی^۵ و برای تجهیزات سخت‌افزاری از گرداننده‌ها^۶ استفاده می‌شود. مسلماً رابط‌های مذکور، دارای تفاوت‌های اساسی هستند. این دسته‌بندی به شناسایی دقیق تر کنشگرها منجر می‌گردد. به عنوان مثال، پاسخ به پرسش‌های زیر در مورد رابط‌های نرم‌افزاری ضروری است:

» رابط ارتباطی سیستم خارجی با کدام فناوری ساخته شده است؟
نمونه پاسخ: وب سرویس.

» دسترسی به آن چگونه امکان‌پذیر است?
نمونه پاسخ: استفاده از نام و نام کاربری.

» چه متدهایی در آن وجود دارد؟ پارامترهای ورود و خروجی هر یک از آن‌ها چیست؟
به عنوان مثالی دیگر، برای شناخت بهتر کاربران، می‌توان پرسش‌های زیر را مطرح کرد:

» تحصیلات کاربر تا چه مقطعی است؟

نمونه پاسخ: متصدیان امور بانکی دارای مدرک کارشناسی یا بالاتر در رشته‌های اقتصاد، آمار یا ریاضی کاربردی هستند.

» آیا کاربر تجربه استفاده از سیستم‌های نرم‌افزاری را دارد؟
نمونه پاسخ: متصدیان امور بانکی دوره ICDL را گذرانده‌اند، اما اغلب تجربه استفاده از سیستم اطلاعاتی تحت وب را ندارند. تجربه آن‌ها تنها محدود به استفاده از اینترنت برای کارهای عمومی است.

» وظیفه اصلی کاربر در سازمان چیست؟ برای انجام آن، سیستم چه کمکی به وی خواهد کرد؟
نمونه پاسخ: وظیفه کاربر ارائه خدمات به مشتری در شعبه بانک است. سیستم باید امکان ارائه خدمات به مشتریان را برای وی فراهم نماید.^۷

1. Scanner
2. Barcode reader
3. Interface
4. User interface (UI)
5. Application programming interface (API)
6. Drivers

۷- معمولاً پاسخ پرسش‌های ارائه شده به صورت مستندات میانی در پروژه‌های نرم‌افزاری نگهداری می‌گردد. در صورت نیاز، می‌توان پاسخ‌ها را در مشخصات کنشگرها مطابق روش گفته شده در فصل بعد مستند نمود.

-۲-۲-۳ - طبقه‌بندی بر اساس اهمیت

در اینجا کنشگرها بر اساس اهمیت در سیستم طبقه‌بندی می‌گردند. دسته‌های این طبقه‌بندی عبارتند از:

- **کنشگر اصلی:** کنشگری که سیستم برای رفع نیاز آن، تولید می‌شود.

- **کنشگر فرعی:** کنشگری که هدف سیستم، رفع نیازهای آن نیست. بلکه برای کارکرد سیستم، لازم است این کنشگر نیز از نرم‌افزار استفاده کند.^۱

به عنوان مثال در سیستم حساب سپرده کوتاه مدت، سیستم برای انجام کارهای متصدیان شعبه تولید می‌گردد. از این رو متصدیان شعبه، کنشگر اصلی محسوب می‌گردد. از طرف دیگر، برای کارکرد سیستم و استفاده متصدیان امور بانکی از آن، کارشناس امور شعب باید شعبه‌ها و پرسنل آن را در سیستم تعریف نماید. در نتیجه کارشناس امور شعب، کنشگر فرعی محسوب می‌گردد.

هدف اصلی از این دسته‌بندی، کمک به اتخاذ تصمیمات درست در انتخاب راه حل مناسب است. از آنجا که در ارائه راه حل‌ها، به ناچار باید از بین چندین راه حل، مناسب‌ترین را انتخاب کرد، این گونه انتخاب‌ها معمولاً با تصمیمات بینایی^۲ همراه هستند. این دسته‌بندی به انتخاب گزینه مناسب کمک می‌کند. بر اساس این دسته‌بندی راه حلی انتخاب می‌گردد که دارای مزایای بیشتری برای کنشگر اصلی باشد. به عنوان مثال متصدی امور بانکی کنشگر اصلی سیستم «حساب سپرده کوتاه مدت» است. در نتیجه راه حل‌های ارائه شده باید به گونه‌ای باشند که این کنشگر در قیاس با سایر کاربران، کارش را با دقت، صحت و راحتی بیشتری انجام دهد.

-۳-۲-۳ - طبقه‌بندی بر اساس روش تعامل با مورد کاربرد

بر خلاف دو طبقه‌بندی قبلی که کنشگرها را بر اساس خصوصیات ذاتی طبقه‌بندی می‌کردن، در این شیوه، کنشگرها بر اساس شکل تعامل با موارد کاربرد طبقه‌بندی می‌گردد. به عبارت دیگر محور اصلی طبقه‌بندی، به جای کنشگر، مورد کاربرد است. در این دسته‌بندی، کنشگرها به شکل زیر طبقه‌بندی می‌گردند:

- **شروع کننده:** کنشگری که یک مورد کاربرد را شروع می‌کند.
- **دربافت کننده:** کنشگری که طی اجرای یک مورد کاربرد، اطلاعاتی را از سیستم دریافت می‌کند.
- **منع خارجی:** سیستم دیگری که اطلاعات لازم در اجرای یک مورد کاربرد را در اختیار سیستم قرار می‌دهد.
- **تسهیل کننده:** کنشگری که کاری را به جای کنشگر دیگری انجام می‌دهد.

۱- در بعضی از مراجع، کنشگر اصلی و کنشگر فرعی، نسبت به هر مورد کاربرد تعیین می‌گردد، یعنی در هر مورد کاربرد، کنشگری که آن را شروع می‌کند، کنشگر اصلی و بقیه کنشگرهای مرتبط با مورد کاربرد، فرعی محسوب می‌گردد. در این کتاب، کنشگر اصلی و فرعی، نسبت به سیستم تعیین می‌شود.

در جدول زیر، نمونه‌هایی از سیستم حساب سپرده کوتاه مدت آورده شده است.

جدول ۶-۱: نمونه طبقه‌بندی بر اساس روش تعامل با مورد کاربرد

کنشگرها	دسته‌بندی
متصلی امور بانکی در ارتباط با موارد کاربرد «واریز به حساب» یا «برداشت از حساب»	شروع کننده
سیستم اطلاعات مشتریان در مورد کاربرد تعریف مشتری	منبع خارجی

پس از شناسایی نوع کنشگر و بر اساس آن، با طرح پرسش‌هایی می‌توان اطلاعات دقیق‌تری از آن به دست آورده تا در تحلیل‌ها و ارائه راه حل‌ها مورد استفاده قرار گیرد. نمونه‌ای از موضوعات قابل پرسش در زیر آورده شده است.

- شروع کننده‌ها: زمان و چگونگی شروع مورد کاربرد، داده‌های مبادله شده بین کنشگر و سیستم و هدف مورد انتظار.
- دریافت کنندگان: چگونگی و زمان برقراری ارتباط، پروتکل ارسال و ساختار اطلاعات ارسالی و چگونگی حصول اطمینان از دریافت اطلاعات توسط دریافت‌کننده.
- تسهیل کننده: احتمال حذف تسهیل کننده در آینده نزدیک و اطلاعاتی که وی برای انجام کار در اختیار دارد.
- منبع خارجی: چگونگی و زمان برقراری ارتباط، قرارداد دریافت و ساختار اطلاعات دریافتی.

۳-۴-۲-۱ - کنشگر زمان^۱

کنشگر زمان بیانگر شرایطی است که سیستم به طور خودکار، کاری را آغاز می‌کند. به عنوان مثال در سیستم حساب سپرده کوتاه مدت، سیستم باید در انتهای هر ماه، به صورت خودکار فعال شود، سود حساب‌های مشتریان را محاسبه کرده، به حساب‌ها واریز کند. این کار در روز و ساعت مشخصی در هر ماه انجام می‌شود که توسط راهبر سیستم مشخص می‌گردد. برای مدل‌سازی این گونه شرایط، از کنشگر زمان استفاده می‌شود.

۳-۳-۳ - تناظر کنشگرهای انسانی و نقش‌های سازمانی

وقتی دو نفر به عنوان کاربر سیستم، کار مشابهی را انجام می‌دهند، بیانگر تنها یک کنشگر در سیستم خواهد بود. به عنوان مثال فرض کنید آقای پویان و خانم مریم، دو کارمند شعبه هستند که سمت آن‌ها متصلی امور بانکی است. با توجه به کارهای مشابه آن‌ها با سیستم، هر دو شخص تنها بیانگر یک کنشگر به نام «متصلی امور بانکی» خواهند بود (شکل ۵-۶).

از طرف دیگر، یک شخص می‌تواند در قالب چندین کنشگر با سیستم کار کند. در شکل ۶-۶ آقای دانیال توأم ن نقش انباردار و راهبر سیستم را ایفا می‌کند.



آقای پویان

خانم مریم

متصدی امور بانکی

متصدی امور بانکی



متصدی امور

افتتاح حساب

بانکی،

شکل ۶-۵: رابطه نقش سازمانی و کنشگر



آقای دانیال به عنوان

آقای دانیال به عنوان

راهبر سیستم

انباردار



راهبر سیستم



انباردار

شکل ۶-۶: یک شخص در قالب دو کنشگر

از موضوعات قبلی نتیجه‌گیری می‌شود که تناظر یک به یک بین کنشگرها و نقش‌های سازمانی وجود ندارد.

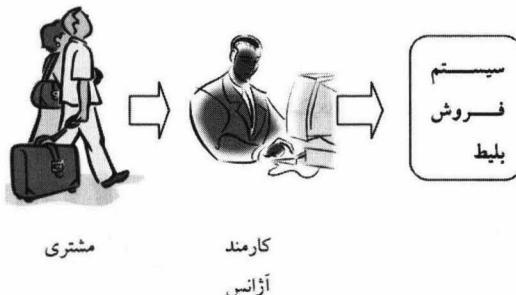
۴-۳- اهمیت کنشگرها

عدم شناسایی یک کنشگر منجر به از دست دادن مجموعه‌ای از قابلیت‌های مورد نیاز در سیستم می‌شود. به عنوان مثال اگر در سیستم حساب سپرده کوتاه مدت، بازرس شعبه به عنوان کنشگر، شناسایی نگردد، مجموعه‌ای از قابلیت‌های سیستم استخراج نخواهد شد.

شناسایی کنشگرها، گامی در جهت شناسایی محدوده سیستم است. فرض کنید که تولید سیستم «فروش بلیط» یک آژانس مسافرتی به تیم واگذار شده است. با هدف شناسایی کنشگرها، سعی کنید به پرسش زیر پاسخ دهید:

آیا مشتری آژانس کنشگر سیستم است؟

پاسخ این پرسش به محدوده سیستم وابسته است. اگر سیستم برای استفاده کارمندان آژانس تولید می‌گردد، پس مشتری از سیستم استفاده نخواهد کرد و کنشگر سیستم محسوب نمی‌گردد.



شکل ۶-۷: رابطه مشتری و کنشگر در سیستم آژانس مسافرتی (۱)

اگر علاوه بر کارمندان آژانس، مشتریان نیز امکان استفاده از سیستم را از طریق اینترنت داشته باشند، آنگاه مشتری نیز کنشگر سیستم خواهد بود.

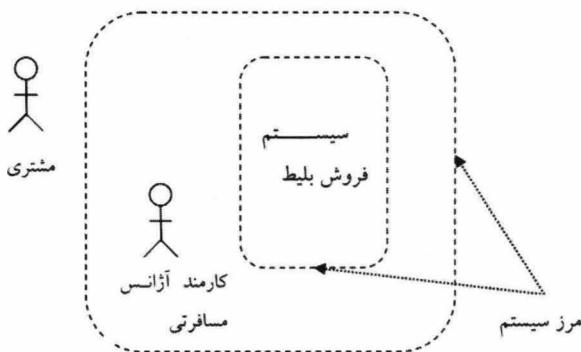


شکل ۶-۸: رابطه مسافر و کنشگر در سیستم آژانس مسافرتی (۲)

انتخاب کنشگر جدید، ممکن است منجر به تغییر نیازمندی‌های کارکردی گردد. به عنوان مثال در سیستم فروش بلیط، هنگامی که سیستم تنها در محل آژانس مورد استفاده قرار می‌گیرد، هزینه بلیط‌ها به صورتی نقدي دریافت می‌گردد. هنگامی که سیستم از طریق اینترنت در اختیار مشتریان آژانس قرار می‌گیرد، وجود قابلیت پرداخت اینترنتی در سیستم برای پرداخت هزینه بلیط ضروری است.

انتخاب کنشگر جدید، علاوه بر تغییر نیازمندی‌های کارکردی سیستم، ممکن است منجر به تغییر نیازمندی‌های غیرکارکردی نیز گردد. به عنوان مثال در سیستم فروش بلیط، هنگامی که سیستم تنها توسط کارمندان آژانس مورد استفاده قرار می‌گیرد، به دلیل تعداد کم کارمندان، استفاده روزمره آنان از سیستم، در دسترس بودن و امکان آموزش کاربران، نیازی به طراحی رابط کاربر ساده و مطابق استانداردهای روز سیستم‌های تحت وب نیست. چرا که کاستی‌های سیستم، با آموزش و استمرار استفاده از آن توسط کاربران به فراموشی سپرده خواهد شد. اما در شرایطی که سیستم توسط مشتریان از طریق اینترنت استفاده می‌گردد، طراحی رابط ساده و مبتنی بر استانداردهای رایج که حداقل پنهانی باند را استفاده نماید، یکی از نیازمندی‌های مهم سیستم خواهد بود.

شکل زیر رابطه کنشگرها و مرز سیستم را در مثال قبلی - سیستم فروش بلیط - نشان می‌دهد. اگر مشتری به عنوان کنشگر سیستم در نظر گرفته شود، مرز سیستم گسترش پیدا می‌کند. از سوی دیگر، با حذف وی نیز مرز سیستم کاهش پیدا خواهد کرد.



شکل ۶-۶: رابطه کنشگر و مرز سیستم

با توجه به نکات مذکور، توصیه می‌گردد که کنشگرهای جدید با دقت بیشتری به سیستم اضافه گرددند.

۴- مورد کاربرد

در این قسمت، مورد کاربرد به عنوان مهم‌ترین جزء مدل مورد کاربرد معرفی می‌گردد.

۴-۱- تعریف مورد کاربرد

مورد کاربرد، بینگر «استفاده»‌ای است که کنشگر از سیستم می‌کند. به عنوان مثال اگر پرسیده شود که «متصدی امور بانکی چه استفاده‌ای از سیستم می‌کند؟»، پاسخ عبارت است از: «تعریف مشتری»، «واریز به حساب»، «برداشت از حساب» و مانند این‌ها. هر یک از استفاده‌های بیان‌شده، یک «مورد کاربرد» سیستم است.

برای استفاده از سیستم، تعامل کنشگر با سیستم ضروری است. به عنوان مثال برای «تعریف مشتری»، متصدی امور بانکی باید نام، نام خانوادگی و شماره ملی مشتری را وارد و گزینه تأیید را انتخاب نماید. پس از آن، سیستم اطلاعات مشتری را ثبت می‌کند. از این رو می‌توان گفت «مورد کاربرد توصیفی از رفتار سیستم است که به شکل مجموعه‌ای از تعاملات بین کنشگر و سیستم بیان می‌شود».

انجام مورد کاربرد، نتیجه‌ای ملموس و ارزشمند را برای کنشگر به همراه دارد. به عنوان مثال، نتیجه مورد کاربرد «واریز به حساب»، واریز مبلغ دریافتی از مشتری به حساب وی توسط متصدی امور بانکی است.

با توجه به مطالب قبلی، «مورد کاربرد» به شکل زیر تعریف می‌گردد:

«مورد کاربرد، مجموعه‌ای از کارهای متوالی سیستم در تعامل با کنشگرها است که نتایج ملموسی را برای آنان به دنبال دارد.

مورد کاربرد در UML به شکل زیر نمایش داده می‌شود.



شکل ۶: نماد مورد کاربرد در UML

۴-۲- انواع مورد کاربرد

موارد کاربرد بر اساس اهمیت‌شان برای کنشگر به دو دسته زیر تقسیم می‌گردند:

﴿ مورد کاربرد اصلی^۱

این دسته از موارد کاربرد، اهداف اصلی تولید سیستم را پوشش می‌دهند. برای مثال در سیستم حساب سپرده کوتاه مدت، موارد کاربرد واریز به حساب یا برداشت از حساب، از این دسته هستند.

﴿ مورد کاربرد فرعی^۲

1. Primary usecase
2. Secondary usecase

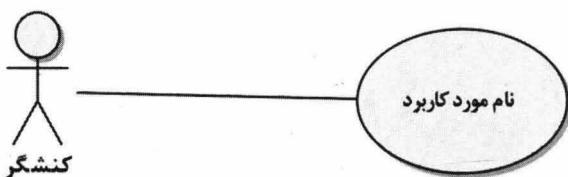
این دسته از موارد کاربرد، پوشش دهنده اهداف اصلی تولید سیستم نیستند، بلکه به دلایلی مانند آماده سازی اطلاعات لازم برای موارد کاربرد اصلی و تکمیل چرخه های کاری وجود دارند. «تعریف شعبه» نامه ای از این دسته از موارد کاربرد است.

در سیستم خودپرداز، مورد کاربرد برداشت و انتقال پول از موارد کاربرد اصلی و اضافه کردن دستگاه خودپرداز به لیست دستگاه های معتبر از موارد کاربرد فرعی است. هم چنین در سیستم فروش بلیط، مورد کاربرد خرید بلیط از موارد کاربرد اصلی و مورد کاربرد ثبت اطلاعات مشتریان از موارد کاربرد فرعی سیستم است، زیرا سیستم برای فروش بلیط ساخته شده است و ثبت اطلاعات مشتریان جزء دلایل ساخت سیستم نیست اما وجود آن برای فروش بلیط ضروری است.

دسته بندی موارد کاربرد به اصلی و فرعی، میزان اهمیت آن ها را مشخص می کند و تعیین کننده اولویت تخصیص منابع، سطوح کیفی و مواردی از این گونه است. در عمل، ممکن است رسمآ اطلاعاتی در مورد اصلی یا فرعی بودن مورد کاربرد در فرایند توسعه نرم افزار نگهداری نگردد، اما این اطلاعات در ذهن تحلیل گر، مدیر پروژه و سایر اعضاء، به صورت ضمنی در تصمیم گیری ها مورد استفاده قرار می گیرند.

۴-۳- مدل سازی ارتباط بین کنشگر و مورد کاربرد

ارتباط کنشگر و مورد کاربرد در UML با رابطه تداعی^۱ نمایش داده می شود. برای نشان دادن اینکه کنشگر، مورد کاربرد را شروع می کند می توان از علامت ناوش^۲ استفاده نمود. ناوش در رابطه تداعی، اطلاع تکمیلی است و اجباری در مشخص کردن آن وجود ندارد.



شکل ۶-۱۱: مدل سازی رابطه کنشگر و مورد کاربرد

۵- سند مشخصات مورد کاربرد

همانطور که گفته شد مورد کاربرد شرح تعامل سیستم و کنشگر است. تدوین گام های تعامل این دو در پروژه های توسعه نرم افزار، ضروری است. «مشخصات مورد کاربرد» سندی است که حاوی شرح تعامل کنشگر و سیستم است. سند مشخصات مورد کاربرد دارای بخش های زیر است^۳:

1. Association
2. Navigation

۳- این سند دارای استاندارد مشخصی نیست. صاحب نظران اجزای مختلفی را برای این سند پیشنهاد کرده اند. در این کتاب از قالب پیشنهادی RUP به عنوان مرجع استفاده شده است.

- نام موردنکاربرد
- شرح مختصر^۱
- گردش رخدادها^۲
- گردش اصلی^۳
- گردش‌های جایگزین^۴
- زیر گردش^۵
- نیازمندی‌های خاص^۶
- پیش شرط‌ها^۷
- پس شرط‌ها^۸
- نقاط گسترش^۹
- اطلاعات تکمیلی^{۱۰}

برای نمونه، مشخصات موردنکاربرد «برداشت از حساب» در زیر ارائه شده است.

جدول ۲-۶: مشخصات موردنکاربرد برداشت از حساب

نام موردنکاربرد: برداشت از حساب	
۱- شرح مختصر	
هدف از این موردنکاربرد، برداشت نقدی از حساب مشتری در شعبه بانک است.	
۲- گردش رخدادها	
۱-۲- گردش اصلی	
○ کاربر گزینه «برداشت» را انتخاب می‌کند.	
○ سیستم فرم «برداشت» را نمایش می‌دهد.	
○ کاربر شماره حساب را وارد می‌کند.*	
○ سیستم اطلاعات حساب به شرح زیر را به کاربر نمایش می‌دهد:	
□ نام صاحب حساب	
□ موجودی	
□ موجودی قابل برداشت	
□ وضعیت حساب	

1. Brief description
2. Flow of events
3. Basic flow
4. Alternatives
5. Subflow
6. Special requirements
7. Precondition
8. Postcondition
9. Extension points
10. Additional information

○ کاربر مبلغ برداشت را وارد و گزینه تأیید را انتخاب می‌کند.

○ سیستم کارهای زیر را انجام می‌دهد:

□ موجودی حساب را به اندازه مبلغ برداشت کسر می‌کند.

□ تراکنش مالی را ثبت می‌کند.

□ پیغام «عملیات با موفقیت انجام شد» را به کاربر نمایش می‌دهد.

○ کاربر گزینه «تأیید» را در فرم پیغام انتخاب می‌کند.

○ سیستم فرم را می‌بندد.

۲-۲-گردش‌های جایگزین

۲-۲-۱-انصراف

کاربر می‌تواند تا قبل از انتخاب گزینه تأیید در فرم برداشت، با انتخاب گزینه انصراف از ادامه عملیات صرف نظر نماید. در این صورت سیستم فرم برداشت را می‌بندد.

۲-۲-۲-نادرست بودن شماره حساب

پس از ورود شماره حساب توسط کاربر، در صورتی که حسابی با شماره وارد شده وجود نداشته باشد:

○ سیستم پیغام «حسابی با این شماره پیدا نشد» را به کاربر نمایش می‌دهد.

○ کاربر گزینه تأیید را در فرم پیغام انتخاب می‌کند.

○ سیستم به فرم برداشت بازمی‌گردد.

۲-۲-۳-عدم امکان برداشت از حساب مسدود

پس از ورود شماره حساب توسط کاربر، در صورتی که حساب در وضعیت «مسدود شده» باشد:

○ سیستم پیغام «این حساب مسدود است و امکان برداشت از آن وجود ندارد» را به کاربر نمایش می‌دهد.

○ کاربر گزینه تأیید را در فرم پیغام انتخاب می‌کند.

○ سیستم به فرم برداشت بازمی‌گردد.

...

...

۳-نیازمندی‌های خاص
ندارد.

۴-پیش‌شرط
ندارد.

۵-پس‌شرط
ندارد.

۶-نقاط گسترش
ندارد.

۷-اطلاعات تکمیلی
ندارد.

در ادامه، اجزای سند مشخصات مورد کاربرد تشریح می‌گردد.

۱-۵- شرح مختصر

این قسمت شامل شرح مختصری درباره مورد کاربرد است که بیان‌گر هدف وجودی و توصیف جایگاه آن در سیستم است.

۲-۵- گردش رخدادها

همانطور که گفته شد مورد کاربرد تعامل بین کنشگر و سیستم است که می‌تواند به صورت محاوره‌ای و به شکل ذیل بیان شود:

کنشگر: [کاری انجام می‌دهد]

سیستم: [در پاسخ کاری را انجام می‌دهد]

کنشگر: [کار دیگری را انجام می‌دهد]

سیستم....

محاوره بین کنشگر و سیستم یک «گردش رخداد» نامیده می‌شود. هر گردش رخداد شامل تنها یک «گردش اصلی» و چندین «گردش جایگزین» است. گردش اصلی بیان‌گر روش عادی محاوره و گردش‌های جایگزین بیان‌گر روش‌های دیگر محاوره و استثنای هستند.

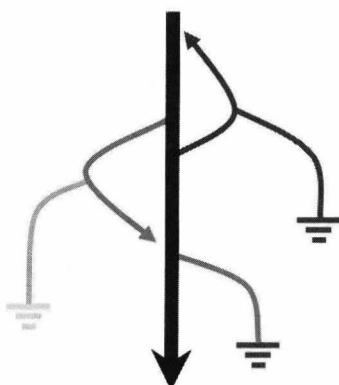
گردش اصلی، روال مطلوب، ایده‌آل یا عادی تعامل کنشگر و سیستم را شرح می‌دهد. در گردش اصلی فرض بر آن است که محاوره به شکل عادی و بدون بروز هیچگونه شرایط خاص یا استثنایی انجام می‌گیرد. هر گردش رخداد حتماً یک و فقط یک گردش اصلی دارد. گردش اصلی، گردش در شرایط «روز خوش^۱» یا «همه چیز خوب^۲» است.

ترتیب گردش اصلی و گردش‌های جایگزین بر روی کارکرد سیستم اثری ندارد. دسته‌بندی گردش رخدادها به گردش اصلی و گردش‌های جایگزین برای خوانایی بیشتر و مدیریت پیچیدگی در مشخصات مورد کاربرد انجام می‌شود.

می‌توان گردش رخداد جایگزین را به عنوان انحرافی از رخداد اصلی دانست که برخی منجر به بازگشت به گردش اصلی و برخی نیز موجب پایان مورد کاربرد می‌شوند. در شکل زیر پیکان مستقیم، گردش اصلی و پیکان‌های منحنی مسیر اجراء‌های جایگزین را نمایش می‌دهند.

1. Happy day

۲- این اصطلاح از یکی از همکاران و دوستان قدیمی به یادگار مانده است.



شکل ۶-۶: ارتباط گردش اصلی و گردش‌های جایگزین

گرددش رخداد ممکن است فاقد گرددش فرعی باشد، هر چند احتمال آن بسیار اندک است. برخی از شرایطی که منجر به ایجاد گرددش جایگزین می‌شوند عبارتند از:

- ﴿ انحراف و تغییر مسیر

در برداشت از حساب، کاربر پس از مشاهده مشخصات حساب می‌تواند از ادامه عملیات صرف نظر نماید و با انتخاب گزینه انصراف از فرم خارج شود.

﴿ استثناهای

در برداشت از حساب، چنانچه حساب مسدود باشد امکان برداشت از آن وجود ندارد و سیستم مانع برداشت از حساب می‌شود یا وقتی که شماره حساب وارد شده نادرست باشد و حسابی با شماره مذکور وجود نداشته باشد، سیستم از ادامه عملیات جلوگیری می‌کند.

زیرگرددش، سازوکاری برای فاکتورگیری بخش‌های مشترک در گرددش رخدادها (اصلی و جایگزین) با هدف افزایش خوانایی و سادگی مشخصات موردکاربرد است.

۵-۳- نیازمندی‌های خاص

نیازمندی‌های غیرکارکردی عمومی سیستم در سند مشخصات تکمیلی^۱ مستندسازی می‌شوند. به عنوان مثال نیازمندی «تمامی کارهای کاربران ثبت گردد»^۲ در سیستم حساب سپرده کوتاه مدت از این نیازمندی‌ها است.

برخی اوقات یک یا چند نیازمندی غیرکارکردی در کل سیستم عمومیت ندارند، بلکه مختص یک موردکاربرد هستند. در این شرایط نیازی به مستندسازی آن‌ها در سند مشخصات تکمیلی نیست. این گونه

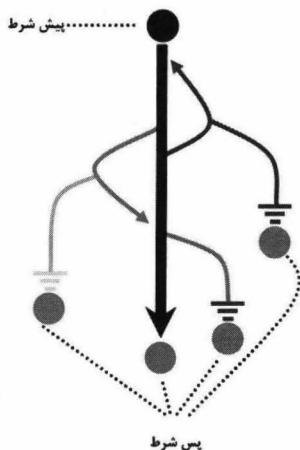
1. Supplementary specifications

۲- به این امر رویدادنگاری یا Logging گفته می‌شود.

نیازمندی‌ها در قسمت «نیازمندی‌های خاص» مستند می‌گردند. به عنوان مثال رمزنگاری داده‌های ردوبدل شده در سیستم حساب سپرده کوتاه مدت جز در موردکاربرد «ورود به سیستم»^۱ ضروری نیست. این نیازمندی در بخش نیازمندی‌های خاص موردکاربرد «ورود به سیستم» مدون می‌گردد.

۴-۵- پیششرط و پسشرط

پیششرط، شرط یا وضعیتی است که قبل از شروع موردکاربرد، سیستم و محیط آن باید در آن شرایط یا وضعیت قرار داشته باشند تا موردکاربرد اجرا شود. پسشرط نیز شرط یا وضعیتی است که بعد از خاتمه اجرای موردکاربرد انتظار می‌رود سیستم در آن شرایط یا وضعیت قرار داشته باشد. در شکل زیر نقاط اعمال و ظهور پیششرط‌ها و پسشرط‌ها در گردش موردکاربرد مشخص شده‌اند.



شکل ۱۳-۶: جایگاه پیششرط و پسشرط در گردش رخدادها

می‌توان مفهوم پیششرط و پسشرط را در قالب یک مثال ریاضی نشان داد. تابع $y = f(x) = |\sqrt{x}|$ را در نظر بگیرید که ریشه دوم یک عدد را محاسبه می‌کند. پیششرط تابع عبارت است از $x \geq 0$ و پسشرط آن عبارت است از $y \geq 0$.

اگر تابع $f(x)$ را مانند جعبه سیاهی در نظر بگیرید که از درون آن اطلاعی وجود ندارد، پیششرط تابع به معنای آن است که در صورت فراخوانی تابع، آیا تابع کار خود را شروع خواهد کرد؟ پسشرط تابع نیز به این معناست که صرف‌نظر از آن چه که در درون تابع می‌گذرد، شرط درستی کارکرد تابع چیست.

برای مثال فرض کنید که برداشت از حساب نیاز به تأیید رئیس شعبه دارد. در سیستم، پس از ثبت فرم برداشت، مدیر باید فهرست آن‌ها را در سیستم مشاهده و تأیید کند. در آن صورت پسشرط موردکاربرد برداشت از حساب به شرح زیر خواهد بود:

«برداشت‌های ثبت‌شده در کارت‌ابل رئیس شعبه قابل مشاهده است.»

در این حالت انتظار داریم پس از اتمام موردکاربرد برداشت از حساب، این فرم در لیست مذکور توسط رئیس شعبه قابل مشاهده و تأیید باشد.

به عنوان مثالی دیگر، فرض کنید که متصدیان شعبه امکان انجام عملیات شعبه‌ای را بدون اعلام «شروع روز کاری» توسط رئیس شعبه ندارند. توجه داشته باشید که آن‌ها قادر به ورود به سیستم هستند ولی امکان انجام عملیات بانکی را ندارند. در این حالت پیش‌شرط کلیه موارد کاربرد مرتبط با عملیات شعبه‌ای مانند واریز و برداشت به شکل زیر خواهد بود:

«روز شعبه آغاز شده باشد.»

حال این پرسش مطرح می‌گردد که چگونه سیستم در وضعیت «آغاز شدن روز شعبه» قرار می‌گیرد؟ این کار با ورود رئیس شعبه به سیستم و انتخاب منوی «اعلام شروع روز» انجام می‌شود. با این کار، سیستم خود را در وضعیت «آغاز شدن روز شعبه» قرار می‌دهد.^۱

۵-۵- نقاط گسترش

نقاط گسترش بیان‌گر رابطه گسترش^۲ یک موردکاربرد با سایر موارد کاربرد است. این نوع رابطه در فصل بعد شرح داده شده است.

۶- اطلاعات تكمیلی

این بخش بیان‌گر سایر اطلاعات مرتبط با موردکاربرد است. از جمله این اطلاعات می‌توان به موارد زیر اشاره نمود:

- » نشانی نمونه اولیه رابط کاربر^۳ یا تصویر آن
- » نشانی نمودار فعالیت در مدل موردکاربرد یا تصویر آن
- » قواعد کسب‌وکار مرتبط

توصیه می‌شود تنها وجود نمونه اولیه رابط کاربر یا نمودار فعالیت در این قسمت از موردکاربرد مشخص گردد. مهم‌ترین مزیت اتخاذ این رویکرد، جلوگیری از تکرار مطالب در مستندات است که کار تغییر و اصلاح آن‌ها را با دشواری همراه خواهد کرد. به عنوان مثال در نظر بگیرید که فرم رابط کاربر در مشخصات موردکاربرد کپی شده باشد. با تغییر فرم به اجبار باید تصویر فرم اصلاح شده در سند مشخصات موردکاربرد نیز اصلاح گردد. این کار زمانبر و خطاز است. تعیین قواعد کسب‌وکار مرتبط با هر موردکاربرد در فصل بعد تشریح می‌گردد.

۱- توجه داشته باشید که این مثال جهت سادگی انتقال مطلب ارائه شده است و در سیستم‌های واقعی ممکن است به شکلی دیگر انجام گردد.

2. Extend

3. User interface prototype

برای مشخص کردن وجود نمونه اولیه رابط کاربر و نمودار فعالیت می‌توان از جدولی به شکل زیر استفاده کرد.

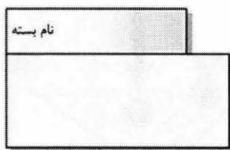
جدول ۳-۶: نمونه جدول مشخصات تکمیلی

بله / خیر [نشانی فرم]	آیا مورد کاربرد دارای نمونه رابط کاربر است؟
بله / خیر [نشانی نمودار فعالیت]	آیا مورد کاربرد دارای نمودار فعالیت است؟

۶- بسته

بسته یکی از عناصر گروه‌بندی در UML است. بسته به مانند پوشه^۱ در سیستم عامل ویندوز عمل می‌کند که ابزاری برای کاهش پیچیدگی و دسته‌بندی فایل‌ها و زیرپوشه‌ها است. بسته‌های مدل مورد کاربرد شامل مجموعه‌ای از موارد کاربرد، کنشگرها، روابط، نمودارها و سایر بسته‌ها هستند. برای اطلاع از نحوه بسته‌بندی مدل مورد کاربرد به فصل بعد مراجعه نمایید.

بسته در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۴-۶: نماد تصویری بسته در UML

۷- سند مورور مدل مورد کاربرد

مدل مورد کاربرد شامل مجموعه‌ای از کنشگرها، موارد کاربرد، مشخصات مورد کاربرد، بسته‌ها و سایر نمودارهاست. به دلیل گسترده‌گی اجزای مدل، لازم است در ابتدا بدون بیان جزئیات مدل، خلاصه‌ای از آن به خواننده ارائه گردد. «سند مورور مدل مورد کاربرد» نقش «خلاصه» مدل مورد کاربرد را ایفا می‌کند. اهمیت این سند با افزایش بزرگ‌سیستم و تعداد موارد کاربرد بیشتر می‌گردد.

با افزایش تعداد موارد کاربرد، مدل مورد کاربرد با استفاده از بسته‌ها به اجزای کوچکتری تقسیم می‌گردد. معمولاً معیار انتخاب بسته‌ها در ذهن مدل‌ساز پنهان می‌ماند. این سند امکان مستندسازی این دانش را نیز فراهم می‌کند.

سند مورور مدل مورد کاربرد شامل بخش‌های کنشگرها، سازماندهی مدل مورد کاربرد و موارد کاربرد مطابق قالب زیر است.

جدول ۶-۴: قالب سند مرور مدل موردنی کاربرد

۱- مقدمه	
...	
۲- کنشگرها	
در این قسمت، کنشگرهای سیستم معرفی می‌گردند.	
۳- سازماندهی مدل موردنی کاربرد	
در این قسمت، چگونگی تقسیم مدل موردنی کاربرد به بخش‌های کوچکتر تشریح می‌گردد.	
۴- موارد کاربرد	
در این قسمت، موارد کاربرد سیستم معرفی می‌گردند.	

۱-۷ - کنشگرها

در این قسمت از سند، نام کنشگرهای همراه توصیف هر یک از آنها بیان می‌شود. برای این کار می‌توان از جدولی به شکل زیر استفاده نمود:

جدول ۵-۶: جدول توصیف کنشگرهای سیستم حساب سپرده

نام کنشگر	شرح مختصر	مورد کاربرد

شرح هر یک از ستون‌ها در زیر آمده است.

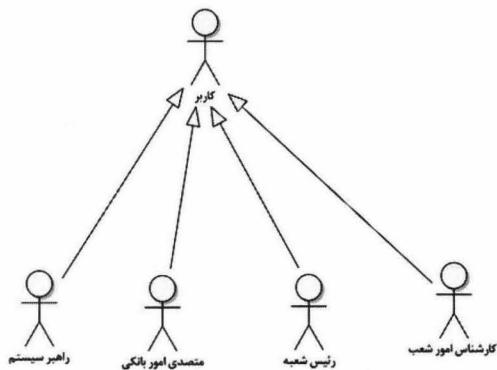
- » نام کنشگر: نام یا عنوان کنشگر
- » شرح مختصر: متن کوتاهی برای معرفی کنشگر
- » مورد کاربرد: فهرست موارد کاربردی که توسط کنشگر انجام می‌شود

جدول ۶-۶: کنشگرهای سیستم حساب سپرده کوتاه مدت

نام کنشگر	شرح مختصر	مورد کاربرد
بانکی متصدی امور		
بانکی متصدی امور	منظور متصدیان امور بانکی مستقر در شعب بانک هستند که وظیفه پاسخ‌گویی و خدمت‌رسانی به ارباب رجوع و مشتریان را به عهده دارند.	افتتاح حساب واریز به حساب برداشت از حساب مسدود کردن حساب رفع مسدودی حساب بستن حساب

در این قسمت روابط بین کنشگرها نیز توصیف می‌گردد. از این‌رو رابطه تعمیم^۱ بین کنشگرها با درج نمودار موردکاربرد یا نمودار کلاس در این قسمت نمایش داده می‌شود.^۲ رابطه تعمیم بین کنشگرها به این معناست که تمامی موارد کاربرد مرتبط با کنشگر پدر^۳ با کنشگرهای فرزند نیز مرتبط هستند.

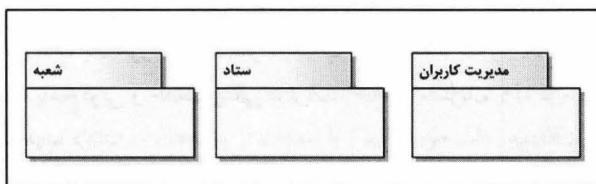
به شکل ۱۵-۶ که از سیستم «حساب سپرده کوتاه مدت» انتخاب شده است، توجه کنید. بر اساس آن، اگر کنشگر «کاربر»، موردکاربرد «ورود به سیستم» و «خروج از سیستم» را اجرا کند، به این معنا است که تمام کنشگرهای فرزند (شامل متصدی امور بانکی، رئیس شعبه، کارشناس امور شعب و راهبر سیستم) نیز قادر به انجام این موارد کاربرد هستند.



شکل ۱۵-۶: نمودار رابطه بین کنشگرهای سیستم حساب سپرده کوتاه مدت

۲-۲- سازماندهی مدل مورد کاربرد

در این بخش بسته‌های موجود در مدل موردکاربرد به همراه معیار و دلایل دسته‌بندی بیان می‌گردد. این بخش عموماً حاوی نمودار بسته^۴ است که تصویری از بسته‌های مدل موردکاربرد را به همراه شرحی از هر یک از بسته‌ها ارائه می‌کند. این موضوع به درک بهتر خواننده از مدل موردکاربرد کمک خواهد کرد. به مثال زیر که از سیستم حساب سپرده کوتاه مدت انتخاب شده است توجه نمایید.



شکل ۱۶-۶: بسته‌های مدل موردکاربرد سیستم حساب سپرده کوتاه مدت

1. Generalization

۲- این نمودار از مدل موردکاربرد در ابزار مدل‌سازی (EA) کمی شود.

3. Parent

4. Package Diagram

﴿ بسته شعبه

موارد کاربردی که در شعبه بانک انجام می‌شوند، در این بسته قرار می‌گیرند.

﴿ بسته ستاد

موارد کاربردی که در ستاد بانک (مدیریت امور شعب) انجام می‌شوند مانند تعریف شعبه در این بسته قرار می‌گیرند.

﴿ بسته مدیریت کاربران

این بسته حاوی موارد کاربردی است که برای مدیریت کاربران و نیز کنترل دسترسی آن‌ها در سیستم وجود دارد. از جمله این موارد کاربرد می‌توان به تعریف کاربر اشاره کرد.

۳-۷ - موارد کاربرد

در این بخش، توصیف مختصری برای هر مورد کاربرد سیستم بیان می‌شود. برای این کار از جدولی مشابه جدول کنشگرها استفاده می‌گردد که به شکل زیر است:

جدول ۷-۶: جدول توصیف موارد کاربرد

نام مورد کاربرد	شرح مختصر	کنشگر

شرح هر یک از ستون‌ها در زیر آمده است.

﴿ نام مورد کاربرد: نام یا عنوان مورد کاربرد

﴿ شرح مختصر: توصیفی از مورد کاربرد شامل یک یا دو پاراگراف

﴿ کنشگر: فهرست کنشگرهايی که مورد کاربرد را اجرا می‌کنند

توجه داشته باشید که ستون آخر از اطلاعات جدول کنشگرها یا نمودار مورد کاربرد قابل استخراج است و وجود آن در اینجا تنها به دلیل کامل بودن اطلاعات هر ردیف از جدول و مراجعه نکردن خواننده به سایر بخش‌های مستند است.

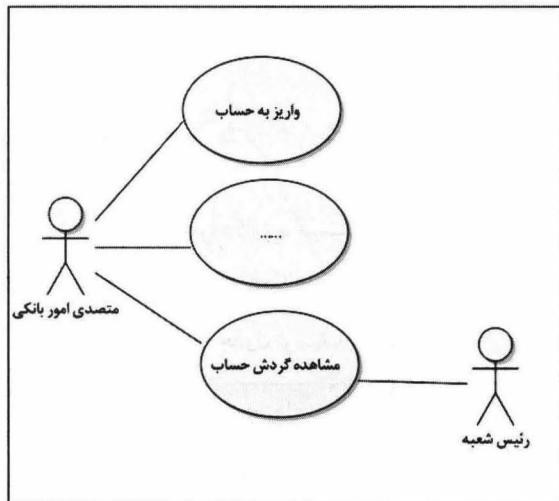
روابط بین موارد کاربرد و کنشگرها به کمک نمودارهای مورد کاربرد در این قسمت نشان داده می‌شوند.^۱

به مثال جدول ۸-۶ که از سیستم حساب سپرده کوتاه مدت انتخاب شده است، توجه نمایید.

۱- این نمودارها از مدل مورد کاربرد در ابزار مدل‌سازی (EA) کپی می‌شوند.

جدول ۸-۶: موردهای کاربرد سیستم حساب سپرده کوتاه مدت

نام مورد کاربرد	شرح مختصر	کنشگر
واریز به حساب	هدف از این مورد کاربرد، واریز نقدی به حساب یک مشتری توسط مراجعه کننده به شعبه است.	متصلی امور بانکی
برداشت از حساب	هدف از این مورد کاربرد، برداشت نقدی صاحب حساب از حساب خود در شعبه است.	متصلی امور بانکی



شکل ۱۷-۶: نمودار نمونه در سند مرور مدل مورد کاربرد

نکته حائز اهمیت آن است که این بخش می‌تواند به تفکیک بسته‌های مدل مورد کاربرد آورده شود. به عنوان مثال موارد کاربرد سیستم حساب سپرده کوتاه مدت می‌توانند در این بخش در سه جدول به تفکیک «شعبه»، «ستاد» و «مدیریت کاربران» بیان شوند.

-۸- کاربرد مدل مورد کاربرد

در فرایندهای توسعه نرم‌افزار مبتنی بر مورد کاربرد^۱، مدل مورد کاربرد یکی از محصولات پراهمیت پروژه است. در این گونه فرایندها، از مدل مورد کاربرد می‌توان برای مقاصد زیر استفاده کرد:

- ﴿ دستیابی به درک مشترکی از رفتارهای سیستم

از آنجا که کاربران و سایر ذینفعان قادر به خواندن، بررسی و اظهارنظر درباره موارد کاربرد هستند، می‌توان بازخورد و نظر آنها را دریافت و پس از بررسی اعمال کرد. با این کار می‌توان به درک مشترکی از رفتار سیستم دست یافت.

۴) طراحی نرم‌افزار

موردنگاری کاربرد بیانگر رفتار بیرونی سیستم از دیدگاه کاربر است. مسئولیت طراح نرم‌افزار این است که با استفاده از ابزارها، فناوری‌ها و روش‌های طراحی، رفتار و ساختار درونی سیستم را به گونه‌ای مشخص کند که قادر به ارائه رفتار بیرونی باشد.

از این رو، موارد کاربرد یکی از مهم‌ترین ورودی‌های طراحی نرم‌افزار است. طراح نرم‌افزار به کمک موارد کاربرد قادر به طراحی پایگاه اطلاعاتی، رابط کاربر و طراحی برنامه کاربردی خواهد بود.

۵) تعریف آزمایش‌ها^۱

در آزمون سیستم به خصوص آزمون کارکردی، تطابق رفتار سیستم با رفتار مورد انتظار انجام می‌گیرد. رفتار مورد انتظار سیستم در قالب موارد کاربرد تدوین می‌گردد.

از این رو موارد کاربرد یکی از ورودی‌های مهم مرحله آزمون سیستم هستند. چرا که جزئیات تعامل، ورودی‌ها و خروجی‌ها، استثنایها و سایر حالت‌ها در آن بیان شده‌اند.

۶) برنامه‌ریزی و تعیین کارها

یکی از روش‌های برآورد زمان و هزینه پروژه، استفاده از موردهای کاربرد است. به عنوان مثال روش «نقاط مورد کاربرد»^۲ یکی از روش‌های شناخته شده در این حوزه است. از موارد کاربرد در اولویت‌بندی کارها، تخصیص کارها به اعضای تیم، شکست کار به کارهای کوچکتر، تعیین خروجی نسخه‌های مختلف سیستم استفاده می‌گردد. هر مورد کاربرد نیاز به تدوین مشخصات، تحلیل، طراحی، پیاده‌سازی و آزمون دارد که وظایف اعضای تیم را مشخص می‌کند.

۷) تهیه راهنمای کاربر

«راهنمای کاربر» به کاربر کمک می‌کند تا از سیستم استفاده نماید. هدف از کار، نتیجه مورد انتظار، گام‌ها، استثنایها و سایر حالت‌ها در مورد کاربرد وجود دارند. کافی است که تهیه‌کننده راهنمای کاربران، روش انجام کار را بر اساس موردنگاری با استفاده از اجزای نرم‌افزار (مانند منوها، فرم‌ها، دکمه‌ها، کلیدهای میان‌بر) بیان کند.

۹- نکات کلیدی

- مدل موردکاربرد، مدلی از نیازمندی‌های کارکردی سیستم است که مبتنی بر موارد کاربرد بیان می‌گردد.
- این مدل شامل کنشگر، موردکاربرد، نمودار موردکاربرد، مشخصات موردکاربرد، بسته و سند مرور مدل موردکاربرد است.
- موردکاربرد، مجموعه‌ای از کارهای متوالی سیستم در تعامل با کنشگرها است که نتایج ملتموسی را برای آنان به دنبال دارد.
- از مدل موردکاربرد می‌توان برای توافق با کاربران، طراحی نرم‌افزار، آزمون سیستم، برنامه‌ریزی و تهیه راهنمای کاربر استفاده کرد.

۱- مراجع

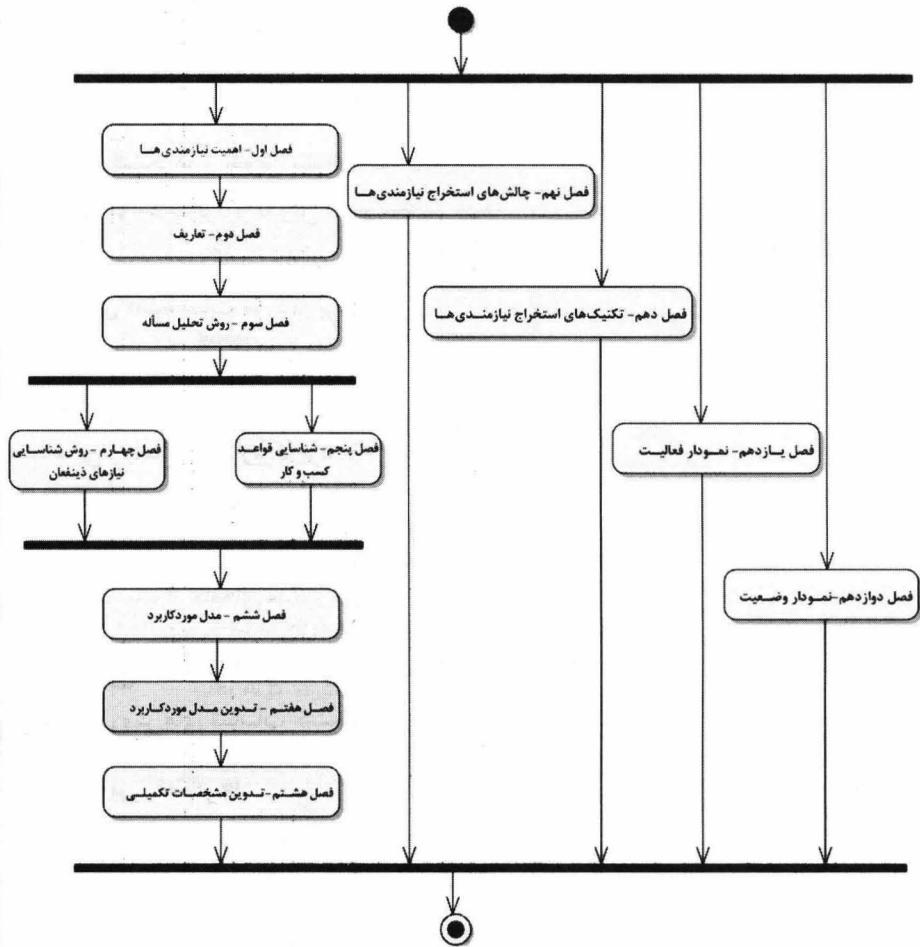
1. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
2. Gause, Donald, and Gerald Weinberg. Exploring Requirements: Quality Before Design. Dorset House Publishing, 1989
3. Leffingwell, Dean, Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
4. OMG. "Software & Systems Process Engineering Meta-Model Specification Version 2." Specifications, Object Management Group, 2007
5. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
6. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>
7. Roques, Pascal. UML in Practice. Wiely, 2004



فصل هفتم - تدوین مدل مورد کاربرد

امروزه مهندسی نیازمندی‌ها بسیار دشوارتر از گذشته است، چرا که تمامی سیستم‌هایی که توصیف آن‌ها ساده بود، مدت‌ها قبل تولید شده‌اند.

تام دی‌مارکو، ارائه‌دهنده روش «تحلیل ساخت‌یافته»



۱- مقدمه

در فصل قبل (فصل ششم)، مدل موردکاربرد و اجزای آن شرح داده شد و بیان شد که از مدل موردکاربرد برای تعریف سیستم استفاده می‌گردد. در این فصل روش تدوین مدل موردکاربرد تشریح می‌گردد. این کار در چند گام انجام می‌شود که در هر گام، بخشی از مدل موردکاربرد کامل می‌گردد. گام‌های تدوین مدل موردکاربرد عبارتند از:

۱. شناسایی کنشگرها و تدوین مشخصات آن‌ها
۲. شناسایی موارد کاربرد و تدوین شرح مختصر آن‌ها
۳. ترسیم «نمودار موردکاربرد» اولیه
۴. طرح‌ریزی «مشخصات موردکاربرد»
۵. تشریح «مشخصات موردکاربرد»
۶. تقسیم موارد کاربرد
۷. شناسایی روابط بین موارد کاربرد
۸. استفاده از سایر نمودارها
۹. سازماندهی مدل موردکاربرد

در این فصل هر یک از گام‌های مذکور بررسی خواهد گردید.

۲- شناسایی کنشگرها و تدوین مشخصات آن‌ها

در این قسمت روش‌های شناسایی کنشگرها و روش توصیف آن‌ها تشریح شده است.

۲-۱- شناسایی کنشگرها

کنشگرها را می‌توان به روش‌های زیر شناسایی نمود:

- پاسخ به پرسش‌های تحلیلی
- ترسیم نمودار متن^۱ برای شناسایی کنشگرها
- بررسی راه حل پیشنهادی^۲

در ادامه هر یک از روش‌های مذکور تشریح می‌گردد.

پاسخ به پرسش‌های تحلیلی برای شناسایی کنشگرها

یکی از راه‌های ساده برای شناسایی کنشگرها، یافتن پاسخ پرسش‌های زیر است:

- چه کسانی اطلاعات سیستم را فراهم (ثبت)، استفاده یا حذف خواهند نمود؟
- چه کسانی برای انجام کارها و وظایفشان به سیستم نیاز دارند؟

1. Context diagram
2. Solution

- چه کسانی کارهای راهبری و نگهداری سیستم را انجام می‌دهند؟
- آیا سیستم با سیستم‌های نرم‌افزاری دیگر تعامل دارد؟
- تجهیزات سیستم مانند سخت‌افزارها کدامند؟

در شکل زیر برخی از پرسش‌های قبلی به صورت تصویری ارائه شده‌اند.



شکل ۷-۱: کنشگرهای بالقوه یک سیستم

نمونه‌ای از پاسخ به پرسش‌های قبلی در سیستم حساب سپرده کوتاه مدت در زیر آمده است.

جدول ۷-۱: نمونه پاسخ‌ها در سیستم حساب سپرده کوتاه مدت

پاسخ	پرسش
متصدی امور بانکی	چه کسانی اطلاعات سیستم را فراهم (ثبت)، استفاده یا حذف خواهند نمود؟
رئیس شعبه متصدی امور بانکی	چه کسانی برای انجام کارها و وظایفشان از سیستم استفاده می‌کنند؟
کارشناس امور شعب کارشناس واحد فناوری اطلاعات(راهبر سیستم).	چه کسانی کارهای راهبری و نگهداری سیستم را انجام می‌دهند؟
سیستم اطلاعات مشتریان	آیا سیستم با سیستم‌های نرم‌افزاری دیگر تعامل دارد؟
کارت‌خوان ^۱	تجهیزات سیستم مانند سخت‌افزارها کدامند؟

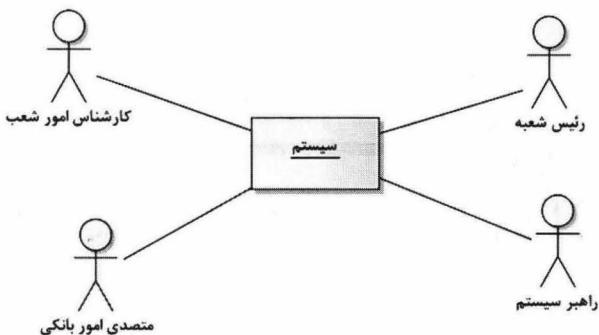
۱- کارت‌خوان در محدوده سیستم نیست و تنها برای نمونه آورده شده است.

ترسیم نمودار متن برای شناسایی کنشگرها

یکی از روش‌های شناسایی کنشگرها، ترسیم نمودار متن است. نمودار متن، نموداری از جنس نمودار گردش داده^۱ است که سیستم را در قالب یک جعبه سیاه در نظر می‌گیرد و ارتباط آن را با دنیای اطراف به شکل تصویری نمایش می‌دهد.

در این نمودار، تمامی اجزای مرتبط با سیستم، شناسایی می‌شوند. می‌توان داده‌های رذوبدل شده بین سیستم و هر کنشگر را به نمودار اضافه کرد.

در شکل ۲-۷، نمونه‌ای از این نمودار برای سیستم حساب سپرده کوتاه مدت ارائه شده است.



شکل ۲-۷: نمودار متن کنشگرها

بررسی راه حل پیشنهادی به کسب و کار برای شناسایی کنشگرها

در این روش فرایند کسب و کار که بر اساس امکانات فناوری اطلاعات پیشنهاد شده است^۲، تحلیل و بررسی می‌گردد. نقش‌های سازمانی که بخش‌هایی از فرایند کسب و کار را با استفاده از سیستم انجام می‌دهند، کاندیدای کنشگر خواهند بود.

منظور از فرایند پیشنهادی، فرایندی است که سازمان پس از نصب و استقرار سیستم نرم‌افزاری (موضوع پروژه)، بر اساس آن کار خواهد کرد. فرایند مذکور توسط تیم توسعه بر اساس امکانات آتی سیستمی که تولید خواهد شد، به ذینفعان پیشنهاد شده و مورد تأیید آن‌ها قرار گرفته است. فرایند پیشنهادی ممکن است به صورت مستندات، مدل یا هر دو در اختیار ذینفعان قرار گیرد.

جدول ۲-۷، بخشی از فرایند «افتتاح حساب» در سیستم حساب سپرده کوتاه مدت است. از آنجا که در این فرایند، متصدی امور بانکی از سیستم استفاده خواهد کرد، کاندیدای یک کنشگر خواهد بود. به دلیل مشابه، چون مشتری در این فرایند از سیستم استفاده نمی‌کند، کنشگر سیستم نخواهد بود.

1. Data flow diagram
2. IT Enabled

جدول ۷-۲: فرایند پیشنهادی افتتاح حساب

فرایند: افتتاح حساب سپرده کوتاه مدت	
○	مشتری به شعبه مراجعه می‌کند.
○	متصدی امور بانکی فرم «افتتاح حساب» را در اختیار مشتری قرار می‌دهد.
○	مشتری پس از تکمیل فرم، آن را به همراه تصویر شناسنامه و کارت ملی و اصل مدارک به متصدی تحويل می‌دهد.
○	متصدی امور بانکی، صحت فرم تکمیلی و مدارک را بررسی می‌کند.
○	متصدی امور بانکی، مشتری را در سیستم تعریف و کد مشتری جدیدی دریافت می‌کند.
○	متصدی امور بانکی، حساب جدیدی را در سیستم برای مشتری افتتاح می‌کند و شماره حساب را روی فرم «افتتاح حساب» درج می‌کند.
.....
.....

برخی اوقات در فرایند پیشنهاد می‌شود که سیستم در زمان یا شرایط خاصی، به صورت خودکار شروع به انجام کاری نماید. در این گونه موارد، شروع خودکار سیستم به صورت کنشگر «زمان» مدل می‌گردد. به عنوان مثال در حساب سپرده کوتاه مدت در انتهای هر ماه، سیستم به صورت خودکار شروع به محاسبه سود می‌نماید و پس از محاسبه، مبلغ سود را به حساب‌ها واریز می‌نماید. در اینجا، زمان کنشگر سیستم است که بیانگر پایان ماه است.

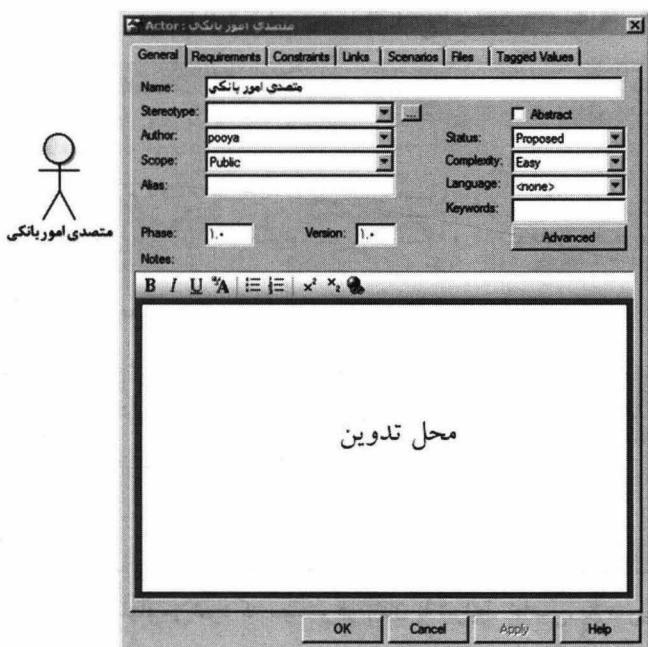
۲-۲- تدوین مشخصات کنشگرها

پس از شناسایی کنشگر، در صورت نیاز شرح مختصری از آن‌ها تهیه می‌گردد. شرح مذکور می‌تواند شامل اطلاعات زیر باشد:

- توصیف کنشگر
- کارهایی که سیستم برای کنشگر انجام می‌دهد
برای تدوین مشخصات کنشگرها می‌توان به شیوه‌های زیر عمل نمود:
- تدوین مشخصات کنشگر در ابزار مدل‌سازی
- تدوین کنشگر در سند مرور مدل موردکاربرد

تدوین مشخصات کنشگر در ابزار مدل‌سازی

یکی از روش‌های ساده تدوین مشخصات کنشگرها، مستندسازی در ابزارهای مدل‌سازی است. با ایجاد کنشگر در ابزار مدل‌سازی، ابزار فرمی را در اختیار کاربر قرار می‌دهد که می‌توان از آن برای تدوین مشخصات کنشگر استفاده نمود. در شکل زیر نمونه‌ای از آن در ابزار EA نشان داده شده است.



شکل ۷-۳. تدوین کنشگر در ابزار EA

این روش برای سیستم‌های کوچک، مناسب است. همچنین در روش‌هایی که تأکید بر حداقل استفاده از ابزار مدل‌سازی و کاهش تعداد مستندات است، این روش کارایی مناسبی دارد.

تدوین کنشگر در سند مرور مدل کاربرد

روش دیگر تدوین مشخصات کنشگرها، مستندسازی آن‌ها در سند مرور مدل کاربرد است. این کار می‌تواند در قالب جدولی به شکل زیر انجام شود.

جدول ۷-۳. قالب تدوین مشخصات کنشگر

ردیف	نام	شرح	موارد کاربرد

- ردیف
 - نام کنشگر
 - شرح مختصر: توصیف کنشگر (در صورت نیاز یا وجود ابهام)
 - موارد کاربرد: فهرست موارد کاربرد قابل انجام توسط کنشگر است.
- در این گام، فهرست موارد کاربرد خالی است و در گام بعدی و پس از شناسایی موارد کاربرد تکمیل می‌گردد.

به عنوان مثال مشخصات کنشگر «متصدی امور بانکی» در زیر آورده شده است:

جدول ۷-۴: نمونه تدوین شرح کنشگر

ردیف	نام	شرح	موارد کاربرد
۱	متصدی امور بانکی	ندارد.	

۳-۲- نکات مهم در شناسایی کنشگرها

در نام‌گذاری کنشگرها به نکات زیر توجه گردد:

- توصیه ۷-۱: نام کنشگر معمولاً اسم مفرد است.
- توصیه ۷-۲: تا حد امکان نام کنشگر از اسمی حوزه مسأله انتخاب گردد.

۳- شناسایی موارد کاربرد و تدوین شرح مختصر آن‌ها

در این گام، موارد کاربرد سیستم شناسایی می‌گردند و هدف از وجود آن‌ها در سیستم در قالب «شرح مختصر» تدوین می‌گردد. در ابتدا، روش‌های شناسایی موارد کاربرد معروفی و در ادامه روش‌های تدوین شرح مختصر آن‌ها بیان می‌گردد.

۳-۱- شناسایی موارد کاربرد

موارد کاربرد به شیوه‌های زیر شناسایی می‌گردند:

- تحلیل کارهای کنشگر
- تحلیل پوشش ویژگی‌های سیستم
- بررسی راهکار پیشنهادی به کسب و کار
- تحلیل ورود و تغییرات داده‌ها و اطلاعات سیستم
- بررسی آغاز و پایان عملیات دوره‌ای
- تحلیل نیازهای زمان راهاندازی، استقرار و نگهداری سیستم

تحلیل کارهای کنشگرها

در این روش، پس از شناسایی کنشگرها با طرح پرسش‌ها و تحلیل پاسخ آن‌ها، موارد کاربرد سیستم شناسایی می‌گردد.

نمونه‌ای از پرسش‌ها در زیر آورده شده است:

- کنشگر، کدام یک از کارهای خود را به کمک سیستم انجام خواهد داد؟
- کنشگر به مطلع شدن از کدام وقایع سیستم نیاز دارد؟
- کنشگر نیاز دارد تا از کدام یک از تغییرات داده‌های سیستم مطلع گردد؟
- کنشگر چه گزارش‌هایی از سیستم دریافت می‌کند؟

در جدول زیر پاسخ‌های مذکور به کنشگر «رئیس شعبه» ارائه شده است. پاسخ‌ها کاندیدای موارد کاربرد در سیستم خواهند بود.

جدول ۷-۵: نمونه پاسخ‌ها در سیستم حساب سپرده کوتاه مدت برای رئیس شعبه

پاسخ	پرسش
<ul style="list-style-type: none"> - بستن حساب شعبه در پایان روز - مشاهده واریز و برداشت‌هایی که باید به تأیید وی رسیده باشند. 	کنشگر کدام یک از کارهای خود را به کمک سیستم انجام خواهد داد؟
<ul style="list-style-type: none"> - موجودی نقدي شعبه در هر لحظه - واریز سود به حساب مشتریان 	کنشگر از کدام واقعی سیستم باید مطلع شود؟
<ul style="list-style-type: none"> - برگشت واریز توسط متصدیان امور بانکی شعبه تحت مدیریت وی - برگشت برداشت توسط متصدیان امور بانکی شعبه تحت مدیریت وی 	کنشگر نیاز دارد تا از کدام تغییرات یا اتفاقات سیستم مطلع گردد؟
<ul style="list-style-type: none"> - گردش موجودی حساب مشتری - گزارش عملکرد متصدیان امور بانکی 	کنشگر چه گزارش‌هایی از سیستم دریافت می‌کند؟

تحلیل پوشش ویژگی‌های سیستم

در این روش با مطالعه ویژگی‌های سیستم در سند چشم‌انداز و بررسی پوشش آن‌ها در راه حل پیشنهادی، موارد کاربرد شناسایی می‌گرددند. در این روش، ابتدا یک ویژگی که در موارد کاربرد پوشش داده نشده است، شناسایی می‌گردد. پس از آن، روش پوشش آن ویژگی در موارد کاربرد فعلی بررسی می‌گردد. در صورت نیاز، مورد کاربرد جدیدی به سیستم اضافه می‌گردد تا ویژگی مذکور تحت پوشش قرار گیرد. به عنوان مثال، در سیستم حساب سپرده کوتاه مدت در سند چشم‌انداز، ویژگی زیر ذکر شده است:

○ مسدود کردن حساب

سیستم باید امکان مسدود کردن یک حساب یا مبلغی از آن را فراهم نماید.

فرض کنید در لحظه تحلیل پوشش ویژگی‌ها، موارد کاربرد زیر در سیستم شناسایی شده باشند:

○ افتتاح حساب

○ واریز به حساب

○ برداشت از حساب

از آنجا که این ویژگی در هیچ یک از موارد کاربرد شناسایی شده، پوشش داده نشده است، باید مورد کاربرد جدیدی به نام «مسدود کردن حساب» به فهرست موارد کاربرد اضافه شود که هدف آن مسدود کردن حساب باشد.

بورسی راه کار پیشنهادی به کسب و کار

در این روش فرایند کسب و کار پیشنهادی بررسی و تحلیل می شود. گام هایی از فرایند که با دخالت نرم افزار انجام می شود، کاندیدای موارد کاربرد خواهد بود.

جدول زیر بخشی از فرایند افتتاح حساب در سیستم حساب سپرده کوتاه مدت است.^۱ در این فرایند، گام های تعریف مشتری و افتتاح حساب، کاندیدای موارد کاربرد هستند. به عبارت دیگر دو مورد کاربرد «تعریف مشتری» و «افتتاح حساب» به فهرست موارد کاربرد اضافه می گردند.

جدول ۶-۷: فرایند پیشنهادی افتتاح حساب

فرایند: افتتاح حساب سپرده کوتاه مدت
○ مشتری به شعبه مراجعه می کند.
○ متصلی امور بانکی فرم «افتتاح حساب» را در اختیار مشتری قرار می دهد.
○ مشتری پس از تکمیل فرم، آن را به همراه تصویر شناسنامه و کارت ملی و اصل مدارک به متصلی تحويل می دهد.
○ متصلی امور بانکی، صحت فرم تکمیلی و مدارک را بررسی می کند.
○ متصلی امور بانکی، مشتری را در سیستم تعریف و کد مشتری جدیدی دریافت می کند.
○ متصلی امور بانکی، حساب جدیدی را در سیستم برای مشتری افتتاح می کند و شماره حساب را روی فرم «افتتاح حساب» درج می کند.
.....

تحلیل ورود و تغییرات داده ها و اطلاعات سیستم

یکی از روش های شناسایی موارد کاربرد، بررسی اشیاء و روش ثبت، تغییر، حذف و مشاهده آنها در سیستم است. در این روش، چرخه زندگی^۲ اشیاء تحلیل می شود تا اطمینان حاصل گردد که عملیات داده ای مانند ثبت، تغییر، حذف و مشاهده شیء در موارد کاربرد وجود دارد. در صورتی که عملیاتی در موارد کاربرد وجود نداشته باشد، روشی برای پوشش آن تعییه می گردد. این کار به شناسایی یک مورد کاربرد جدید یا تغییر یک مورد کاربرد موجود، منجر می گردد.

به عنوان مثال، اعلامیه مسدود کردن حساب از مراجع مختلف مانند سایر دوایر بانک، مراجع قضایی یا انتظامی دریافت می گردد. اطلاعات اعلامیه مسدودی باید در سیستم ثبت، ویرایش، مشاهده و حذف گردد. ممکن است با اعلامیه دیگری، اعتبار اعلامیه مسدودی حساب باطل شود و حساب از مسدودی خارج گردد. در صورتی که هر یک از عملیات مذکور در سیستم دیده نشده باشد، ناگزیر انجام آنها منجر به تغییر موارد کاربرد فعلی می گردد یا شناسایی موارد کاربرد جدیدی را به دنبال دارد.

۱- یادآوری می گردد که ممکن است فرایند مذکور، علاوه بر مستندسازی، مدل سازی نیز گردد.

بررسی آغاز و پایان عملیات دوره‌ای

برخی از موارد کاربرد مرتبط با شروع و پایان دوره‌های زمانی تکرارپذیر در چرخه زندگی سیستم یا کسب و کار هستند. در این روش دوره‌های زمانی مهم در کسب و کار یا سیستم شناسایی می‌گردد و کارکردهای سیستم در ابتداء، انتهایا در طول دوره بررسی می‌گردد. به عنوان مثال سیستم حساب سپرده کوتاه مدت دارای دوره‌های تکرارپذیر روز، ماه و سال است. در ابتداء و انتهای هر یک از دوره‌ها، سیستم باید عملیات تجمعی داده‌ها، پاکسازی داده‌های بی‌صرف یا نهایی‌سازی دوره مرتبط را انجام دهد. اعلام آغاز روز و اعلام پایان روز (در شعبه) و اعلام آغاز سال مالی و اعلام پایان سال مالی (در ستاد) از این نوع موارد کاربرد هستند.

از نمونه‌های دیگر این گونه دوره‌های زمانی می‌توان به سال مالی در سیستم حسابداری و ترم در سیستم ثبت‌نام دانشگاه اشاره نمود.

تحلیل نیازهای زمان راه‌اندازی، استقرار و نگهداری سیستم

یکی از روش‌های شناسایی موارد کاربرد، تحلیل نیازهایی است که زمان راه‌اندازی، استقرار و استفاده از سیستم به آن‌ها نیاز است. این گونه موارد کاربرد معمولاً در ابتداء فراموش می‌گردند و هر چه زمان آزمون سیستم نزدیک‌تر می‌شود، بیشتر به چشم می‌آیند.

در سیستم حساب سپرده کوتاه مدت با افتتاح حساب و ورود مبلغ افتتاحیه و انجام عملیات واریز و برداشت، موجودی حساب تغییر می‌کند. در این سیستم، نیازی به تغییر موجودی حساب مشتری توسط کاربر نیست. اما زمان استقرار سیستم این نیاز مطرح خواهد شد که اطلاعات حساب‌های مشتریان فعلی، چگونه به سیستم وارد گردد. این نیاز ممکن است منجر به تعریف مورد کاربرد جدیدی علاوه بر افتتاح حساب گردد.

موارد کاربرد «تعریف کاربر»، «تخصیص حقوق دسترسی» نیز از این گونه موارد کاربرد هستند.

۲-۳- تدوین شرح مختصر موارد کاربرد

همان طور که در فصل قبل گفته شد، شرح مختصر، یک یا دو پاراگراف کوتاه است که بیانگر هدف وجود مورد کاربرد در سیستم و ارزش افزوده آن برای کاربر است.

برای مستندسازی شرح مختصر می‌توان به شیوه‌های زیر عمل نمود:

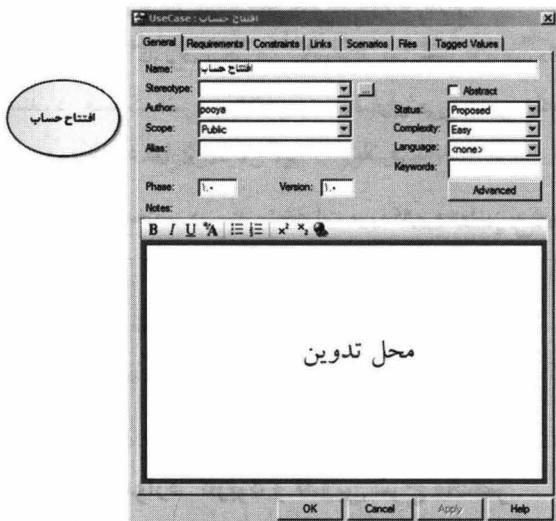
- تدوین شرح مختصر مورد کاربرد در ابزار مدل‌سازی
- تدوین شرح مختصر مورد کاربرد در سند مرور مدل مورد کاربرد
- تدوین شرح مختصر مورد کاربرد در سند مشخصات مورد کاربرد

تدوین شرح مختصر مورد کاربرد در ابزار مدل‌سازی

یکی از روش‌های تدوین شرح مختصر مورد کاربرد، مستندسازی آن‌ها در ابزارهای مدل‌سازی است. با

ایجاد موردکاربرد در ابزار مدلسازی، ابزار فرمی را در اختیار کاربر قرار می‌دهد که می‌توان از آن برای تدوین شرح مختصر استفاده کرد.

در شکل ۴-۷ نمونه‌ای از آن در ابزار EA نشان داده شده است. این روش برای سیستم‌های کوچک بسیار مناسب است. همچنین در روش‌هایی که تأکید بر حداکثر استفاده از ابزار مدلسازی و کاهش تعداد مستندات است، این روش کارایی مناسبی دارد.



شکل ۴-۷: تدوین موردکاربرد در ابزار EA

تدوین شرح مختصر مورد کاربرد در سند مرور مدل مورد کاربرد

روش دیگر تدوین شرح مختصر موردکاربرد، مستندسازی آن در سند مرور مدل مورد کاربرد است. برای مستندسازی شرح مختصر می‌توان از جدولی مانند جدول زیر استفاده کرد.

جدول ۷-۷: قالب تدوین شرح مختصر مورد کاربرد

ردیف	نام موردکاربرد	شرح مختصر	کنشگرها
○			

○ ردیف

○ نام موردکاربرد

○ شرح مختصر

○ کنشگرها: فهرست کنشگرهایی که با موردکاربرد در ارتباط هستند.

به عنوان مثال شرح مختصر موردکاربرد «واریز به حساب» در زیر آورده شده است.

جدول ۷-۸: نمونه تدوین شرح مختصر مورد کاربرد

ردیف	نام مورد کاربرد	شرح مختصر	کنشگرها
۱	واریز به حساب	هدف از این مورد کاربرد، واریز وجه به حساب سپرده کوتاه مدت مشتری است.	متصلی امور بانکی

تدوین شرح مختصر مورد کاربرد در سند مشخصات مورد کاربرد

در این روش پس از شناسایی موارد کاربرد، برای هر یک، سند «مشخصات مورد کاربرد» ایجاد و شرح مختصر آن‌ها در همان سند، در قسمت «شرح مختصر» مستندسازی می‌گردد. به عنوان مثال، جدول زیر بخشی از سند مشخصات مورد کاربرد «واریز به حساب» را نشان می‌دهد.

جدول ۷-۹: شرح مختصر در سند مشخصات مورد کاربرد «واریز به حساب»

۱-شرح مختصر
هدف از این مورد کاربرد، واریز وجه به حساب سپرده کوتاه مدت مشتری است.
۲-گردش رخدادها
.....

۳-۳- نکات مهم در شناسایی موارد کاربرد و تدوین شرح مختصر

- توصیه ۷-۳: نام مورد کاربرد باید توصیف کوتاهی از تعامل کنشگر با سیستم باشد.
- توصیه ۷-۴: نام مورد کاربرد باید منعکس کننده هدف کنشگر از تعامل با سیستم باشد.
- توصیه ۷-۵: تا حد امکان کاربر باید بتواند مورد کاربرد را به عنوان یک کار در حوزه کاری و تخصصی خود شناسایی نماید.
- توصیه ۷-۶: نام مورد کاربرد باید از دیدگاه کنشگر بیان گردد.
- توصیه ۷-۷: نام مورد کاربرد باید فعل معلوم یا عبارت اسمی باشد. به عبارت دیگر با فعل شروع و به اسم ختم گردد.
- توصیه ۷-۸: اسم به کار رفته در نام مورد کاربرد باید مفرد باشد مگر در مورد اقلامی که بیانگر «أنواع» باشند.
- توصیه ۷-۹: شرح مختصر یک یا حداقل دو پاراگراف ساده باشد.
- توصیه ۷-۱۰: شرح مختصر تنها هدف کنشگر از اجرای مورد کاربرد یا دلیل وجود مورد کاربرد در سیستم را بیان می‌کند. این قسمت شامل اطلاعاتی مانند فرایندهای کسب و کار و محیط بیرونی نرم‌افزار نیست، مگر آن که شرح آن‌ها به بیان هدف مورد کاربرد کمک کند.
- توصیه ۷-۱۱: از بیان گام‌های انجام مورد کاربرد در شرح مختصر پرهیز نماید، چون در بخش گردش رخدادها ذکر خواهد گردید.

۴- توصیم نمودار مورد کاربرد اولیه

در این قسمت و پس از شناسایی کنشگرها و موارد کاربرد، «نمودار مورد کاربرد» در ابزار مدل‌سازی ایجاد و کنشگرها و موارد کاربرد به همراه روابط بین آن‌ها در آن نمایش داده می‌شود.

۵- طرح ریزی مشخصات مورد کاربرد

در این مرحله، گردش اصلی از مشخصات مورد کاربرد تکمیل می‌گردد. همچنین در این مرحله، نام گردش‌های جایگزین مشخص می‌گردد و تکمیل آن‌ها در مرحله بعد انجام می‌شود.

۵-۱- نکات مهم در طرح ریزی مشخصات مورد کاربرد

در این قسمت، نکاتی که باید در طرح ریزی مورد کاربرد رعایت گردن، آورده شده‌اند.

نکات عمومی در گردش رخدادها

- توصیه ۷-۱۲: از بیان کلماتی که باعث ابهام یا نامشخص شدن محدوده سیستم می‌گردد، پرهیز کنید. اصطلاحات «...»، «مانند»، «امثالهم»، «غیره» و «مربوطه» از این دست هستند.
- توصیه ۷-۱۳: تعامل کنشگر و سیستم به صورت جملات معلوم بیان گردن. جملاتی که با فاعل شروع و به یک فعل ختم می‌گرددند.
- توصیه ۷-۱۴: از بیان اتفاقات و فرایندهای خارج از نرم‌افزار در گردش رخدادها خودداری نمایید. مورد کاربرد مربوط به کارهای نرم‌افزار است.
- توصیه ۷-۱۵: از به کاربردن نام کنشگر انسانی در مشخصات مورد کاربرد و بهویژه در گردش رخدادها دوری کنید، چرا که در صورت تغییر کنشگر، اعمال تغییرات بسیار وقت‌گیر خواهد بود. کنشگر هر مورد کاربرد به راحتی از روی نمودار مورد کاربرد یا سند مرسول مدل مورد کاربرد قابل شناسایی خواهد بود. نام سایر انواع کنشگر (سایر سیستم‌ها و تجهیزات) در مورد کاربرد آورده می‌شود.
- توصیه ۷-۱۶: مورد کاربرد، بیان گر طراحی نرم‌افزار شامل طراحی بانک اطلاعاتی و طراحی برنامه کاربردی نیست. از طراحی کردن در لابهای مورد کاربرد پرهیز کنید.
- توصیه ۷-۱۷: مورد کاربرد به زبان کاربر نوشته می‌شود نه زبان برنامه‌سازی. در نتیجه سعی نکنید متن مورد کاربرد شبیه یکی از زبان‌های برنامه‌نویسی شود.

شروع و خاتمه مورد کاربرد در گردش اصلی

- توصیه ۷-۱۸: نحوه‌ی شروع مورد کاربرد باید در ابتدای گردش اصلی آورده شود.
- توصیه ۷-۱۹: نحوه‌ی اتمام مورد کاربرد باید در انتهای گردش اصلی آورده شود.

نمایش اطلاعات مورد کاربرد در گردش اصلی

- توصیه ۷-۲۰: در گردش اصلی باید داده‌هایی که کنشگر وارد می‌کند، مشخص شوند.
- توصیه ۷-۲۱: اجباری یا اختیاری بودن داده‌های ورودی کاربر مشخص گردد. پیشنهاد می‌شود این کار با استفاده از علامت ستاره (*) در جلوی حوزه اطلاعاتی (فیلد) اجباری انجام گردد.
- توصیه ۷-۲۲: در صورتی که حوزه اطلاعاتی دارای مقدار پیش‌فرض باشد، مقدار پیش‌فرض مشخص گردد.
- توصیه ۷-۲۳: در صورتی که حوزه اطلاعاتی از لیستی انتخاب می‌شود، علاوه بر مشخص شدن انتخاب از لیست، موجودیت اصلی و اقلام آن که در لیست نمایش داده می‌شوند، مشخص گردد.
- توصیه ۷-۲۴: در صورتی که مقدار حوزه اطلاعاتی بر اساس انتخاب حوزه دیگری مشخص می‌گردد، رابطه بین آن دو مشخص شود.
- توصیه ۷-۲۵: در صورتی که حوزه اطلاعاتی از بین چند گزینه انتخاب می‌گردد، گزینه‌ها مشخص شوند.
- توصیه ۷-۲۶: از به کار بردن مفاهیم فناوری مانند فناوری‌های رابط کاربر دوری کنید، مگر آن که خیلی عمومی باشد و به فناوری خاصی اشاره نکند.

۲-۵- قدوین پیغام‌ها

پیغام‌های سیستم به کاربر را می‌توان به روش‌های زیر مدون کرد:

- » الف: نوشتن متن کامل پیغام در گردش اصلی یا گردش‌های جایگزین در این روش، متن پیغام در گردش اصلی یا گردش‌های جایگزین مشخصات مورد کاربرد، نوشته می‌شود.
- » به مثال زیر در مورد کاربرد «برداشت از حساب» توجه نمایید. ابتدا کاربر شماره حساب را وارد می‌کند، سپس سیستم، مشخصات حساب را به کاربر نمایش می‌دهد. اگر شماره حساب نادرست باشد، سیستم پیغامی را به کاربر نمایش می‌دهد:
پس از ورود شماره حساب، در صورتی که شماره حساب نادرست باشد، سیستم پیغام «شماره حساب مورد نظر در سیستم وجود ندارد» را به کاربر نمایش می‌دهد.
- » ب: نام بردن از پیغام و ارجاع به کد پیغام در فهرست پیغام‌های سیستم در این روش، پیغام‌های سیستم در مستند جداگانه‌ای به نام «پیغام‌ها^۱» به شکل جدول زیر مدون می‌گردد.

جدول ۷-۱۰: قالب تدوین پیغام‌ها

کد	متن پیغام

○ کد: کد پیغام

○ متن پیغام

در ادامه، مثال قبلی (برداشت از حساب) بازنویسی شده است.

پس از ورود شماره حساب، در صورتی که شماره حساب موجود نباشد، سیستم پیغامی مبنی بر پیدا نشدن حساب به کاربر نمایش می‌دهد (پیغام ۴۵). کد ۴۵ اشاره به پیغامی در فهرست پیغام‌ها دارد که مانند جدول ۱۱-۷ نوشته شده است.

جدول ۷-۱۱: پیغام ۴۵ در سند «پیغام‌ها»

کد	متن پیغام
۴۵	شماره حساب مورد نظر در سیستم وجود ندارد

استفاده از فهرست پیغام‌ها در تدوین آن‌ها به دلایل زیر توصیه می‌گردد:

- در فناوری‌های جدید، معمولاً پیغام‌ها در فایل‌هایی خارج از کد^۱ برنامه نگهداری می‌شوند.
این روش امکان انتقال یکباره پیغام‌ها به فایل‌های مذکور را فراهم می‌کند.
- این شیوه، شناسایی پیغام‌های تکراری یا مشابه را ساده‌تر می‌کند.
- در این روش، متن پیغام‌ها می‌تواند مستقل و جدای از محتوای موارد کاربرد به تأیید کاربران رسانده شود.

۳-۳- تدوین قواعد کسب و کار

قواعد کسب و کار را می‌توان به شیوه‌های زیر در مورد کاربرد مدون کرد:

﴿الف: نوشتن متن کامل قاعده کسب و کار در گام‌های گردش رخدادها

در این روش، قاعده کسب و کار به طور کامل در متن مشخصات مورد کاربرد آورده می‌شود. مثال زیر از مورد کاربرد برداشت از حساب انتخاب شده است.

سیستم موجودی قابل برداشت حساب را با فرمول زیر محاسبه و به کاربر نمایش می‌دهد:

موجودی قابل برداشت برابر است با موجودی حساب منهای حداقل موجودی حساب
منهای جمع مبالغ مسدود شده

1. Source

۲- فایل‌های با پسوند Property یا مشابه آن.

﴿ ب: نوشتن قواعد در قسمت جداگانه‌ای از مشخصات موردکاربرد و ارجاع به آن در این روش، قسمتی با نام قواعد کسب‌وکار به مشخصات مورد کاربرد اضافه می‌گردد. قواعد کسب‌وکار مرتبط به هر موردکاربرد در این بخش از مشخصات آن‌ها نوشته می‌شوند. در مشخصات موردکاربرد، بدون ذکر متن قاعده، تنها به آن ارجاع داده می‌شود.

این روش به دلیل تکرار شدن قواعد مشترک بین چندین موردکاربرد توصیه نمی‌گردد.

﴿ ج: ارجاع به قاعده موجود در سند قواعد کسب‌وکار
در این روش، قواعد کسب‌وکار در سند قواعد کسب‌وکار مدون می‌گردند و در مشخصات موردکاربرد، بدون ذکر متن قاعده، تنها به آن ارجاع داده می‌شود. مثال زیر، بازنویسی شده مثال قبلی در موردکاربرد برداشت از حساب است.

موردکاربرد: برداشت از حساب

سیستم، موجودی قابل برداشت حساب را محاسبه (قاعده ۱۱: محاسبه مبلغ قابل برداشت) و به کاربر نمایش می‌دهد.

قاعده ۱۱ به قاعده‌ای در سند قواعد کسب‌وکار اشاره می‌کند که به شکل زیر نوشته شده است:
سند قواعد کسب‌وکار:

قاعده ۱۱: محاسبه مبلغ قابل برداشت

موجودی قابل برداشت برابر است با موجودی حساب منهای حداقل موجودی حساب
منهای جمع مبالغ مسدود شده

برای سیستم‌های متوسط و بزرگ، استفاده از سند قواعد کسب‌وکار با وجود دشوار شدن مطالعه و درک موردکاربرد به دلایل زیر توصیه می‌گردد:

○ سند قواعد کسب‌وکار جدا از نحوه و محل استفاده از آن‌ها در سیستم، می‌تواند به تأیید کاربران و ذینفعان برسد.

○ تیم طراحی نرم‌افزار می‌تواند با درک راه حل و سیستم، با تمرکز و مرور این سند، برای هر قاعده روش طراحی خود را انتخاب کند. به عنوان نمونه در مثال بالا، تیم طراحی تصمیم می‌گیرد که در صورت تغییر موجودی حساب، هنگام ذخیره آن، داده‌های مذکور را محاسبه و در بانک اطلاعاتی نگهداری کند و در سناریوهای مورد استفاده، تنها داده‌های از قبل محاسبه شده را نمایش دهد.

○ این شیوه شناسایی قواعد تکراری یا مشابه را آسان‌تر می‌کند.

۶- تشریح مشخصات موردکاربرد

در این گام از تدوین مشخصات موردکاربرد، گردش رخدادهای جریان‌های جایگزین و زیرگردش‌ها و همچنین نیازمندی‌های خاص، پیششرط و پسشرط کامل می‌گردد.

۶- نکات مهم در تشریح مشخصات مورد کاربرد

در این جا مجموعه‌ای از نکاتی که هنگام تشریح مشخصات مورد کاربرد باید مدنظر قرار گیرد، فهرست شده است.

نکات مرتبط با گردش‌های جایگزین

- توصیه ۲۷-۷: نحوه شروع گردش جایگزین باید در ابتدای گردش جایگزین آورده شود. نقطه شروع گردش جایگزین می‌تواند نقطه‌ای در گردش اصلی یا یکی دیگر از گردش‌های جایگزین باشد.
- توصیه ۲۸-۷: نحوه اتمام گردش جایگزین باید در انتهای گردش جایگزین آورده شود. نقطه اتمام گردش جایگزین می‌تواند نقطه‌ای در گردش اصلی یا گردش‌های جایگزین باشد. ممکن است نقطه اتمام گردش جایگزین پایان مورد کاربرد باشد.
- توصیه ۲۹-۷: برای مشخص کردن نقطه شروع یا پایان گردش‌های جایگزین در گردش اصلی، می‌توانید از برچسب‌گذاری در گردش اصلی استفاده کنید.

نکات مرتبط با نیازمندی‌های خاص

- توصیه ۳۰-۷: هر نیازمندی خاص دارای عنوان و متنی برای توصیف خود است.
- توصیه ۳۱-۷: نیازمندی‌هایی که در کل سیستم مشترک هستند باید در سند مشخصات تکمیلی مستند گردند.
- توصیه ۳۲-۷: تعداد اجرای مورد کاربرد در یک بازه زمانی، همزمانی استفاده توسط کنشگران و مسائل امنیتی از مهم‌ترین نیازمندی‌های خاص مورد کاربرد هستند.

نکات مرتبط با پیش‌شرط و پس‌شرط

- توصیه ۳۳-۷: پیش‌شرط‌ها بیان‌گر موارد کاربردی که باید قبل از مورد کاربرد اجرا شوند، نیستند.
- توصیه ۳۴-۷: پس‌شرط‌ها بیان‌گر موارد کاربرد که می‌توانند بعد از مورد کاربرد اجرا گردند، نیستند.
- توصیه ۳۵-۷: پیش‌شرط رخداد شروع کننده مورد کاربرد نیست.

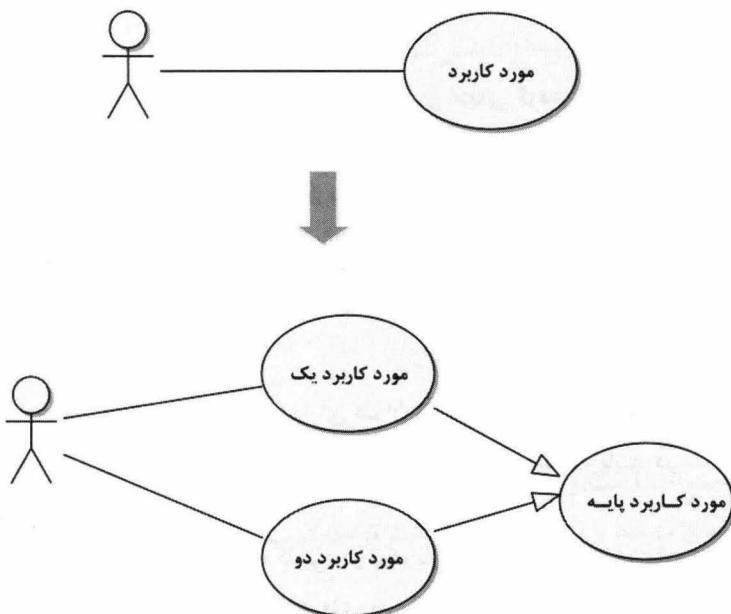
۷- تقسیم موارد کاربرد

وقتی که گردش رخدادهای مورد کاربرد تکمیل گردد، امکان دارد مورد کاربرد به دو یا چند مورد کاربرد دیگر تقسیم (تبدیل) گردد. در این قسمت، دو حالتی که منجر به تقسیم موارد کاربرد می‌گردد- تقسیم عمودی و

تقسیم افقی - مورد بررسی قرار گرفته است.

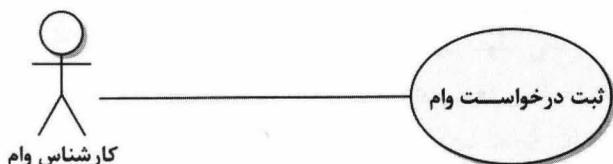
۱-۱- تقسیم عمودی مورد کاربرد

تقسیم عمودی زمانی رخ می‌دهد که گرددش رخدادهای مورد کاربرد به دلیل وجود قاعده‌های کسب و کار یا وجود تنوع در موجودیت^۱ اصلی مورد استفاده در آن به صورت مجموعه‌هایی چند بخشی ظاهر می‌شوند. در این وضعیت مورد کاربرد به دو یا چند مورد کاربرد دیگر تبدیل می‌گردد.



شکل ۷-۵: تقسیم عمودی مورد کاربرد

به عنوان مثال مورد کاربرد «ثبت درخواست وام» را در نظر بگیرید که هدف آن ثبت درخواست وام متقاضیان وام در سیستم است.



شکل ۷-۶: نمودار مورد کاربرد درخواست وام

مشخصات آن به شرح زیر است.

موردکاربرد: درخواست وام

۱-شرح مختصر

....

۲-گردش رخدادها

۱-۱-گردش اصلی

...

کاربر مشخصات متقاضی را به شرح زیر وارد می‌کند:

○ اگر متقاضی حقوقی باشد:

● نام *

● کد اقتصادی *

○ اگر متقاضی حقیقی باشد

● نام *

● نام خانوادگی *

● شماره ملی *

...

کاربر مشخصات تحويل دهنده و زمان آن را به شرح زیر وارد می‌کند:

○ تاریخ *

○ تحويل دهنده *

○ شماره تماس *

...

کاربر اطلاعات اسناد تحولی را به شرح زیر وارد می‌کند:

○ اگر متقاضی حقوقی باشد:

● شماره صمانت نامه بانکی *

● تاریخ *

● ..

○ اگر متقاضی حقیقی باشد:

● شماره سند ملکی *

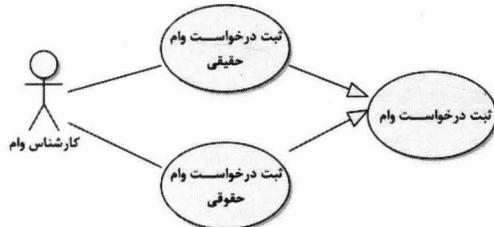
● ..

....

....

گام‌های موردکاربرد از سه بخش مختلف تشکیل شده است: بخش‌هایی که بین دو نوع متقاضی حقیقی و حقوقی مشترک هستند، بخش‌هایی که خاص متقاضی حقیقی هستند و بخش‌هایی که مختص متقاضی حقوقی‌اند. در این حالت می‌توان موردکاربرد را به دو قسمت جدا تقسیم نمود: بخشی که به متقاضی حقیقی اختصاص دارد و بخشی که به متقاضی حقوقی تعلق دارد.

بنابراین موردکاربرد به دو موردکاربرد دیگر یعنی «ثبت درخواست وام حقیقی» و «ثبت درخواست وام حقوقی» تقسیم شده است. با توجه به رابطه تعمیم^۱ بین موارد کاربرد باید موردکاربرد دیگری برای بیان وجه اشتراک دو موردکاربرد مذکور ایجاد گردد. این موردکاربرد اصلی یعنی «ثبت درخواست وام» است.^۲

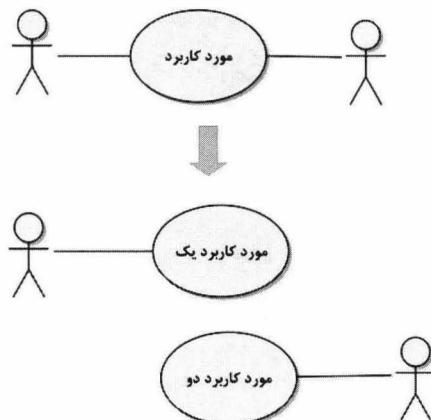


شکل ۷-۷: نمودار پس از تقسیم موردکاربرد درخواست وام

۲-۷- تقسیم افقی موردکاربرد

تقسیم افقی هنگامی اتفاق می‌افتد که گردش رخدادهای موردکاربرد به دلایل زیر به دو یا چند بخش مختلف قابل تقسیم باشند:

- وجود فاصله زمانی در انجام گردش رخدادها
- انجام بخش‌های موردکاربرد توسط دو یا چند کنشگر متفاوت در این حالت موردکاربرد به دو یا چند موردکاربرد دیگر شکسته می‌گردد.



شکل ۷-۸: تقسیم افقی موردکاربرد

به عنوان مثال موردکاربرد «تهیه و ارسال فهرست حضور و غیاب» را در نظر بگیرید که هدف آن در سیستم کارت ساعت، تهیه فهرست حضور و غیاب ماهانه و ارسال آن به سیستم کارگرینی است.

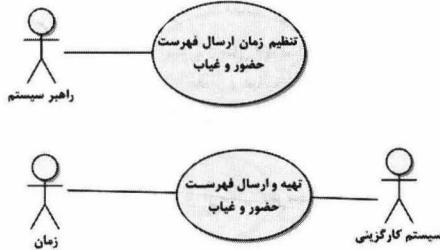


شکل ۹-۷: نمودار تهیه و ارسال فهرست حضور و غیاب

مشخصات آن به شرح زیر است.

مورد کاربرد: تهیه و ارسال فهرست حضور و غیاب	
۱-شرح مختصر	
....
۲-گردش رخدادها	۲-گردش رخدادها
۱-۱-گردش اصلی	۱-۱-گردش اصلی
» کاربر گزینه ارسال فهرست حضور و غیاب را انتخاب می کند.	» کاربر گزینه ارسال فهرست حضور و غیاب را انتخاب می کند.
» ...	» ...
» کاربر اطلاعات زمان ارسال را به شرح زیر وارد می کند	» کاربر اطلاعات زمان ارسال را به شرح زیر وارد می کند
○ روز *	○ روز *
○ ساعت *	○ ساعت *
» کاربر گزینه «تأیید» را انتخاب می کند.	» کاربر گزینه «تأیید» را انتخاب می کند.
» سیستم روز و ساعت تهیه و ارسال فهرست حضور و غیاب را ثبت می کند.	» سیستم روز و ساعت تهیه و ارسال فهرست حضور و غیاب را ثبت می کند.
» در روز و ساعت مشخص شده سیستم فعال می گردد.	» در روز و ساعت مشخص شده سیستم فعال می گردد.
» سیستم فهرست حضور و غیاب یک ماهه شامل اطلاعات زیر را تهیه می کند.	» سیستم فهرست حضور و غیاب یک ماهه شامل اطلاعات زیر را تهیه می کند.
○ تاریخ	○ تاریخ
○ کد پرستنی	○ کد پرستنی
○ نوع مرخصی	○ نوع مرخصی
○ ساعت/تاریخ از	○ ساعت/تاریخ از
○ ساعت/تاریخ تا	○ ساعت/تاریخ تا
.....

با دقت به مورد کاربرد بالا مشخص می گردد که فاصله زمانی بین دو قسمت از مورد کاربرد وجود دارد. در قسمت اول مورد کاربرد، زمان ارسال تنظیم می گردد و در قسمت دوم، فهرست مرخصی ها تهیه و به سیستم کارگزینه ارسال می شود. بین این دو قسمت، فاصله و تأخیر زمانی وجود دارد. بنابراین مورد کاربرد به دو مورد کاربرد یعنی «تنظیم زمان ارسال فهرست حضور و غیاب» و «تهیه و ارسال فهرست حضور و غیاب» تقسیم می گردد.



شکل ۷-۱۰: نمودار پس از تقسیم مورد کاربرد

۸- شناسایی روابط بین موارد کاربرد

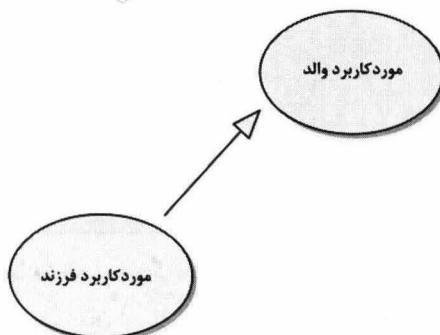
در این گام از تدوین مدل و مشخصات مورد کاربرد، روابط بین موارد کاربرد شناسایی می‌گردد.
به طور کلی سه رابطه بین موارد کاربرد وجود دارد:

- تعمیم
- شامل^۱
- گسترش^۲

۱- رابطه تعمیم

رابطه تعمیم در UML به شکل زیر تعریف می‌گردد:
«رابطه‌ای بین دو عنصر که ساختار، رفتار و معنای^۳ یکی به نام پدر به عنصر دیگر به نام فرزند متقل می‌گردد».

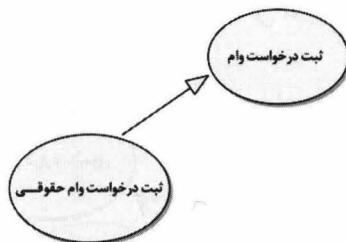
رابطه تعمیم در موارد کاربرد بر اساس همین تعریف شکل می‌گیرد. هر گاه دو یا چند مورد کاربرد از نظر معنا و رفتار انجام شده در گردش‌های آن‌ها مشابه باشند، تشکیل یک ساختار وراثتی خواهند داد.



شکل ۸-۱: رابطه تعمیم بین موارد کاربرد

1. Include
2. Extend
3. Semantic

به عنوان مثال، مورد کاربرد «ثبت درخواست وام حقیقی» و «ثبت درخواست وام حقوقی» دارای مفهوم و گرددش رخدادهای مشابهی هستند. می‌توان با تعریف مورد کاربرد «ثبت درخواست وام»، رابطه تعمیم بین آن‌ها را به شکل زیر برقرار کرد.



شکل ۱۲-۷: نمونه رابطه تعمیم بین موارد کاربرد

۱-۱-۸- مستندسازی رابطه تعمیم

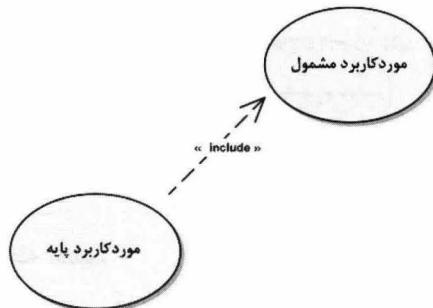
- تدوین موارد کاربردی که رابطه تعمیم با یکدیگر دارند، به شکل زیر انجام می‌گردد:
- ﴿ گرددش اصلی مورد کاربرد پدر، به صورت کلی نوشته می‌شود به گونه‌ای که گام‌های اصلی و مشترک موارد کاربرد فرزند را بیان کند.
 - ﴿ برای گرددش‌های جایگزین مشترک، پیش شرط‌ها، پس شرط‌ها و نیازمندی‌های خاص مشترک بین تمامی موارد کاربرد فرزند، روش‌های زیر وجود دارد:
 - تنها در مورد کاربرد پدر مستندسازی گرددند.
 - هم در مورد کاربرد پدر و هم در موارد کاربرد فرزند مستندسازی گرددند.
 - تنها در موارد کاربرد فرزند مستندسازی گرددند.
 - استفاده از روش اول باعث می‌گردد برای فهم و استنباط موارد کاربرد فرزند، مورد کاربرد پدر نیز مطالعه گردد. روش دوم منجر به افزونگی، تکرار مطالب و سختی اعمال تغییرات خواهد گردید. در روش سوم، شناسایی قسمت‌های مشترک بین موارد کاربرد فرزند، توسط خواننده انجام می‌شود و در آن‌ها مستند نگردیده است. برای پرهیز از پیچیدگی و افزایش خوانایی مشخصات مورد کاربرد، توصیه می‌شود از روش سوم استفاده گردد.

۲-۸- رابطه شمول

رابطه‌های شمول و گسترش همواره محل مناقشه و اختلاف نظر صاحب‌نظران بوده است. در (Adolph, 2002) گفته شده است که:

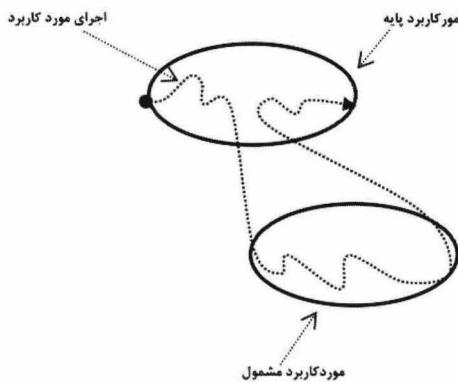
«اگر در کنفرانسی با موضوع نیازمندی‌ها شرکت کردید و تصمیم گرفتید کنفرانس را به هم بریزید، کافی است تا وقتی را برای سخنرانی درخواست کنید. به پشت تریبون بروید و از حضار پرسید که «تفاوت بین رابطه شمول و گسترش چیست؟» پس از آن با خیال راحت از کنفرانس خارج شوید. چون کنفرانس با مباحثه و مجادله حضار به هم می‌ریزد.»

رابطه شمول نوعی رابطه وابستگی^۱ است. رابطه وابستگی در UML به شکل زیر تعریف می‌گردد: «رابطه وابستگی رابطه‌ای بین دو عنصر است که تغییر یکی (عنصر مستقل)، ممکن است منجر به تغییر دیگری (عنصر وابسته) گردد». این رابطه بیان‌گر وابستگی بین دو مورد کاربرد است که یکی مورد کاربرد پایه^۲ و دیگری مورد کاربرد مشمول^۳ نامیده می‌شود.



شکل ۷-۷: رابطه شمول در بین موارد کاربرد

این رابطه بیان می‌کند وقتی که مورد کاربرد پایه به نقطه‌ای در گام‌های خود رسید، مورد کاربرد دیگری را فراخوانی می‌کند و مورد کاربرد مشمول اجرا می‌شود. پس از اجرای آن، اجرای مورد کاربرد پایه ادامه می‌یابد. در شکل ۷-۷ نحوه اجرای مورد کاربرد پایه که با مورد کاربرد دیگری رابطه شمول دارد، ارائه شده است.

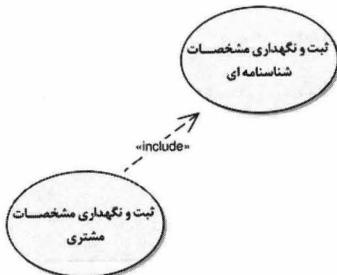


شکل ۷-۷: نحوه اجرای موارد کاربرد دارای رابطه شمول

به عنوان مثال فرض کنید که مورد کاربرد «ثبت و نگهداری مشخصات شناسنامه‌ای» مورد کاربرد مستقلی

1. Dependency
2. Base use case
3. Inclusion usecase

باشد. در موردکاربرد «ثبت و نگهداری مشخصات مشتری» لازم است که مشخصات شناسنامه‌ای مشتری نیز در سیستم ثبت گردد. از آنجا که این بخش در موردکاربرد قبلی وجود دارد، تنها کافی است که در نقطه‌ای از موردکاربرد «ثبت و نگهداری مشخصات مشتری» به موردکاربرد دوم یعنی «ثبت و نگهداری مشخصات شناسنامه‌ای» ارجاع داده شود.



شکل ۱۵-۷: مثال رابطه شمول

رابطه شمول را می‌توان در حالت‌های زیر استفاده نمود:

- فاکتورگیری بخش مشترک و تکراری در دو یا چند موردکاربرد
- جداسازی بخشی از یک موردکاربرد پیچیده با هدف ساده‌سازی و افزایش خوانایی آن

۱-۲-۸ مستندسازی رابطه شمول

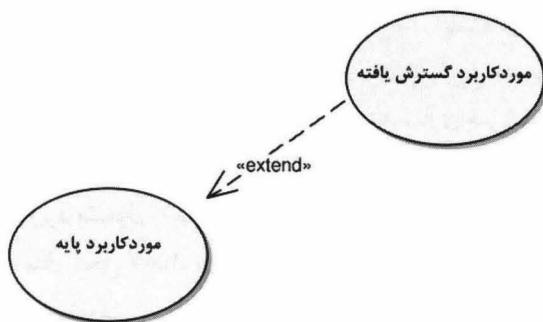
مشخصات موردکاربرد مشمول مانند هر موردکاربرد دیگری مستندسازی می‌شود. در موردکاربرد پایه، نقطه‌ای که موردکاربرد مشمول باید فراخوانی و اجرا گردد، به صورت زیر مدون می‌گردد:
[جمله‌ای که هدف موردکاربرد مشمول را بیان دارد] (شامل: [نام موردکاربرد مشمول]).
 به عنوان نمونه، برای مثال شکل ۱۵-۷، می‌توان مشخصات موردکاربرد ثبت و نگهداری مشخصات مشتری را به شکل زیر تدوین نمود:

موردکاربرد: ثبت و نگهداری مشخصات مشتری
۱-شرح مختصر
....
۲-گردش رخدادها
....
۲-۱-گردش اصلی
....
کاربر با انتخاب گزینه «مشخصات عمومی» مشخصات شناسنامه‌ای مشتری را در سیستم وارد می‌کند <u>(شامل: ثبت و نگهداری مشخصات شناسنامه‌ای).</u>
....

موردکاربرد: ثبت و نگهداری مشخصات شناسنامه‌ای	
۱-شرح مختصر
.....
۲-گردش رخدادها
.....
۲-۱-گردش اصلی
موردکاربرد وقتی شروع می‌شود که کاربر اقدام به ورود مشخصات شناسنامه‌ای می‌کند.
کاربر مشخصات شخص شامل موارد زیر را وارد می‌کند:
○ نام

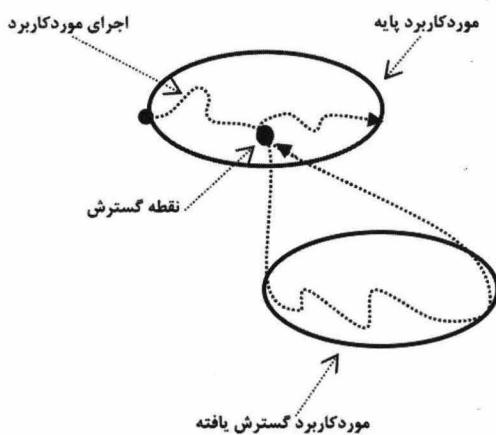
۳-۸- رابطه گسترش

رابطه گسترش مانند رابطه شمول، نوعی رابطه وابستگی است. این رابطه بیان‌گر وابستگی بین دو موردکاربرد است که یکی موردکاربرد پایه و دیگری موردکاربرد گسترش‌یافته^۱ نامیده می‌شود. رابطه گسترش، موردکاربرد گسترش‌یافته را به موردکاربرد پایه متصل و مرتبط می‌کند.



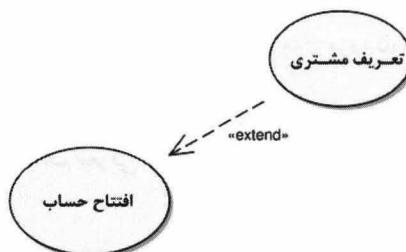
شکل ۱۶-۷: رابطه گسترش در بین موارد کاربرد

این رابطه بیان می‌کند که وقتی که موردکاربرد پایه به نقطه‌ای در گام‌های خود رسید، موردکاربرد گسترش‌یافته شرایطی را بررسی می‌کند و در صورت برقرار بودن شرایط، اجرا می‌گردد. پس از اجرای موردکاربرد گسترش‌یافته، اجرای موردکاربرد پایه ادامه می‌یابد. در شکل ۱۷-۷، نحوه اجرای موردکاربرد پایه که با موردکاربرد دیگری رابطه گسترش دارد، ارائه شده است.



شکل ۷-۷: نحوه اجرای موارد کاربرد دارای رابطه گسترش

به عنوان مثال فرض کنید که در موردکاربرد افتتاح حساب، کاربر شماره ملی صاحب حساب را وارد می‌کند. در صورتی که قبلًا مشتری ای با این شماره وجود نداشته باشد، کاربر انتظار دارد تا با انتخاب گزینه‌ای به فرم تعریف مشتری رفته، بعد از آن به فرم اول برگشته و حساب را افتتاح نماید. در این مثال موردکاربرد «تعریف مشتری»، موردکاربرد «افتتاح حساب» را گسترش می‌دهد یا «تعریف مشتری» گسترشی به «افتتاح حساب» است.



شکل ۱۸-۷: مثال رابطه گسترش

توجه داشته باشید که موردکاربرد تعریف مشتری، موردکاربرد مستقلی است که می‌تواند توسط کنشگر مستقیماً اجرا گردد.

۱-۳-۸- مستندسازی رابطه گسترش

در بخش «نقاط گسترش» موردکاربرد پایه مشخص می‌کنیم که کدام موارد کاربرد، موردکاربرد مذکور را گسترش می‌دهند. علاوه بر عنوان، نقطه و شرایط اجرای موردکاربرد گسترش یافته مشخص می‌گردد. گاهی پارامترهای مورد استفاده موردکاربرد گسترش یافته و اتفاقات پس از برگشت از موردکاربرد گسترش یافته، تعیین می‌گردد.

برای ارجاع به موردکاربرد گسترش یافته از روش پیشنهادی زیر می‌توان استفاده کرد.

[جمله‌ای که هدف موردکاربرد گسترش یافته را بیان دارد] (گسترش: [نام موردکاربرد گسترش یافته]).

در مثال قبل، کاربر با انتخاب گزینه «جدید» به فرم تعریف مشتری رفته، مشتری جدیدی را تعریف کرده، پس از آن حساب را افتتاح می‌کند. پس از تعریف مشتری و برگشت به فرم افتتاح حساب، شماره ملی، نام، نام خانوادگی و تصویر شخص در فرم نمایش داده می‌شود. مشخصات موردکاربرد افتتاح حساب به شکل زیر تدوین می‌گردد:

موردکاربرد: افتتاح حساب
۱-شرح مختصر
....
۲-گردش رخدادها
.....
۶-نقاط گسترش
۶-۱-تعریف مشتری
کاربر می‌تواند با انتخاب گزینه «جدید»، مشتری جدیدی را در سیستم تعریف نماید (توسعه: تعریف مشتری). پس از ثبت مشتری، شماره ملی، نام، نام خانوادگی و تصویر وی نمایش داده می‌شود.

۹- استفاده از سایر نمودارها

در این قسمت، کاربرد نمودار فعالیت^۱ و نمودار وضعیت^۲ در مدل موردکاربرد، تشریح می‌گردد.

۹-۱- نمودار فعالیت

اجزای نمودار فعالیت در فصل یازدهم معرفی شده‌اند. نمودار فعالیت در مدل موردکاربرد در شرایط زیر استفاده می‌گردد:

- مدل‌سازی یک موردکاربرد پیچیده
- مدل‌سازی پوشش فرایند کسب و کار با موارد کاربرد در ادامه شرایط مذکور تشریح گردیده است.

۹-۱-۱- مدل‌سازی موردکاربرد پیچیده

در تدوین مشخصات موردکاربرد، ابتدا گردش اصلی بیان می‌گردد و در ادامه روش‌های دیگر انجام کار یا حالت‌های استثناء توصیف می‌شوند. در نتیجه، برای فهم موردکاربرد، خواننده باید ترکیب منطقی گردش اصلی و گردش‌های جایگزین را بر اساس نقطه ورود گردش‌های جایگزین در گردش اصلی در ذهن تصور نماید. به عنوان مثال، موردکاربرد برداشت از حساب را در نظر بگیرید:

1. Activity Diagram
2. State Diagram

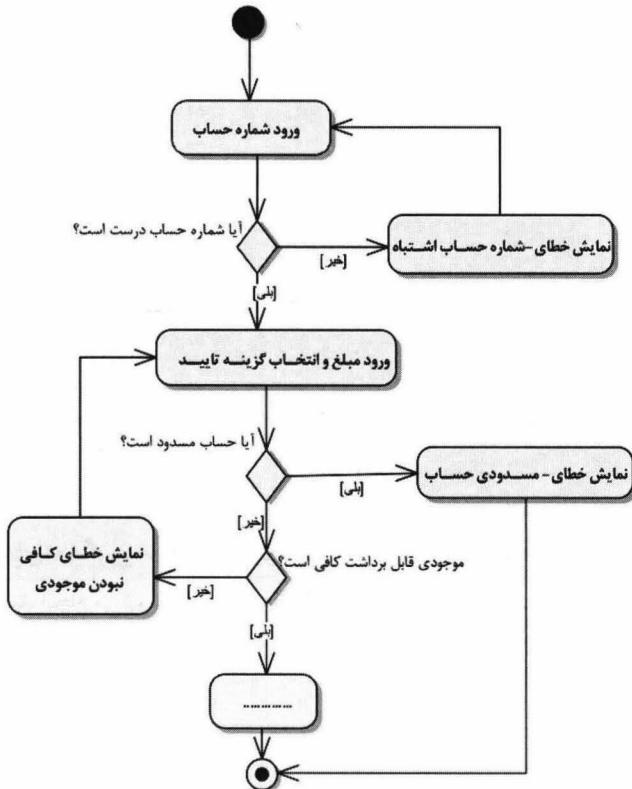
۱-شرح مختصر
۲-گردش رخدادها
۲-۱-گردش اصلی
۲-۲-گردش‌های جایگزین
۲-۲-۱- کافی نبودن موجودی قابل برداشت	پس از انتخاب گزینه تأیید،
۲-۲-۲- مسدود بودن حساب	پس از انتخاب گزینه تأیید،.....
۲-۲-۳- نادرست بودن شماره حساب	پس از ورود شماره حساب توسط کاربر،.....

تمامی گردش‌های جایگزین مانند «کافی نبودن موجودی قابل برداشت» و «مسدود بودن حساب» پس از انتخاب گزینه تأیید انجام می‌شوند و گردش‌های دیگر مانند «نادرست بودن شماره حساب»، پس از ورود شماره حساب انجام می‌گردد. به عبارت دیگر، توالی انجام کار در این مورد کاربرد به شکل زیر است:

- ... ○
ورود شماره حساب
- کنترل درستی شماره حساب
- اگر درست نبود، اعلام پیغام ...
- ... ○
انتخاب گزینه تأیید
- کنترل کافی بودن موجودی قابل برداشت
- اگر کافی نبود، اعلام پیغام ...
- کنترل مسدودی حساب
- اگر حساب مسدود بود، اعلام پیغام ...
- ○

در مشخصات مورد کاربرد، مشخص نیست که با انتخاب گزینه تأیید، کارهای «کافی نبودن موجودی قابل برداشت» یا «مسدود بودن حساب» به چه ترتیبی انجام می‌شوند (کدامیک زودتر از دیگری انجام می‌شود).

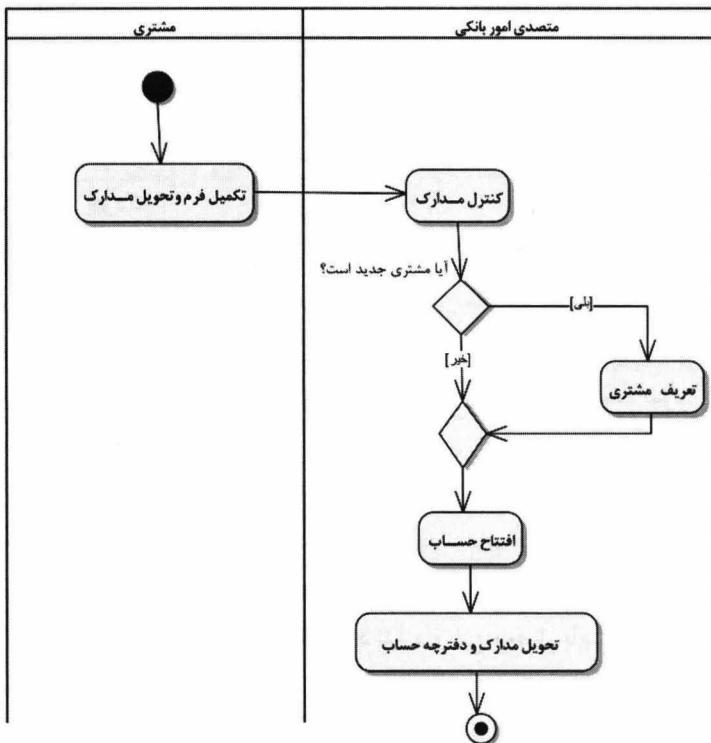
تصور توالی انجام مورد کاربرد با ترکیب گردش اصلی و گردش‌های جایگزین، برای موارد کاربرد پیچیده‌تر به سادگی امکان‌پذیر نیست. از نمودار فعالیت برای تصویرسازی گام‌های موارد کاربرد پیچیده می‌توان استفاده نمود.



شکل ۷-۱۹: نمودار فعالیت مورد کاربرد برداشت از حساب

۲-۱-۹- مدل‌سازی پوشش فرایند کسب کار با موارد کاربرد

راه حل پیشنهادی به یک فرایند کسب‌کار، اغلب منجر به تعریف مجموعه‌ای از موارد کاربرد می‌گردد. به عنوان مثال فرایند افتتاح حساب سپرده کوتاه مدت در بانک، در قالب موارد کاربرد تعریف مشتری و افتتاح حساب انجام می‌شود. به علاوه در افتتاح حساب، کارهای غیرنرم‌افزاری دیگری مانند تکمیل و امضاء اوراق توسط مشتری انجام می‌شود. برای ارائه و تفهیم راه حل به کاربران، لازم است تا تصویری از کل فرایند افتتاح حساب شامل قسمت‌های نرم‌افزاری و غیرنرم‌افزاری به آن‌ها نشان داده شود. برای این کار، نمودار فعالیت، یکی از بهترین گزینه‌هاست.



شكل ٧-٢٠: نمودار فعالیت فرایند افتتاح حساب در شعبه

- فعالیتهای دستی

منظور کارهایی است که بدون کمک گرفتن از نرم افزار انجام می‌شوند، مانند تحویل مدارک (شامل کپی شناسنامه، کارت ملی و اصل شناسنامه و کارت ملی) توسط مشتری به متخصصی امور بانکی.

فعالیت‌های ترکیبی (دستی و نرم‌افزار) ○
منظور فعالیت‌هایی است که بخشی از آن به کمک نرم‌افزار و بخش دیگر بدون دخالت آن انجام می‌گیرد، مانند تطابق تصویر امضا (که نرم‌افزار آن را نشان می‌دهد) با امضای مشتری در فرم برداشت کاغذی (که به صورت چشمی و انسانی انجام می‌گردد).

○ فعالیت‌های مکانیزه

منظور فعالیت‌هایی است که با تعامل کاربر در نرم‌افزار انجام می‌شود. مثلاً واریز به حساب که کاربر اطلاعات آن را وارد می‌کند و سیستم سایر کارها مانند ثبت اطلاعات واریز، افزایش موجودی و صدور سند حسابداری را انجام می‌دهد.

○ فعالیت‌های خودکار

منظور فعالیت‌هایی است که کاربر در انجام آن دخالتی نداشته و معمولاً کارهایی است که سیستم بر اساس تنظیمات قبلی انجام می‌دهد، مثل محاسبه سود تمامی حساب‌ها در انتهای ماه و واریز آن به حساب مشتریان.

برای بیان انواع کارهای قبلی در نمودار فعالیت از روش‌های زیر می‌توان استفاده نمود:

○ استفاده از رنگ‌ها

در این روش، برای هر نوع فعالیت، رنگی مشخص می‌گردد و فعالیت‌ها در نمودار، رنگ‌آمیزی می‌گردند. ممکن است انتخاب رنگ، با انتخاب پس‌زمینه^۱ نیز همراه باشد. از آنجا که رنگ‌آمیزی، روشنی دیداری است، بسیار مناسب است. در چاپ‌های سیاه و سفید نمودارها، امکان تفکیک رنگ‌ها وجود ندارد و این معضل باعث دشواری بررسی آن‌ها می‌گردد.

○ استفاده از کلیشه^۲

در این روش برای هر نوع فعالیت، کلیشه‌ای تعیین می‌گردد. هر کلیشه می‌تواند دارای شمايل^۳ خاص خود باشد.

○ استفاده از جداساز^۴

در این روش به کمک جداساز، نمودار به چند بخش تقسیم می‌گردد که هر بخش بیانگر یک نوع فعالیت است (حداکثر سه بخش). فعالیت‌ها به تفکیک نوع خود، در بخش مربوطه چیده می‌شوند. این روش به دلیل جداسازی و تفکیک نوع فعالیت‌ها به صورت دیداری، روش مناسبی است، اما معمولاً به کارگیری آن منجر به نمودارهای بسیار پیچیده می‌گردد.

۲-۹ - نمودار وضعیت

اجزای نمودار وضعیت در فصل دوازدهم معرفی شده است. این نمودار، ابزاری برای تحلیل سیستم است و یکی از کاربردهای آن، مدل‌سازی حالت‌های یک شیء مهی سیستم است. ابتدا وضعیت‌های شیء و گذارهای^۵ تبدیل وضعیت‌ها به یکدیگر شناسایی و مدل می‌گردد. وضعیت‌ها و رخدادهای منجر به تغییر

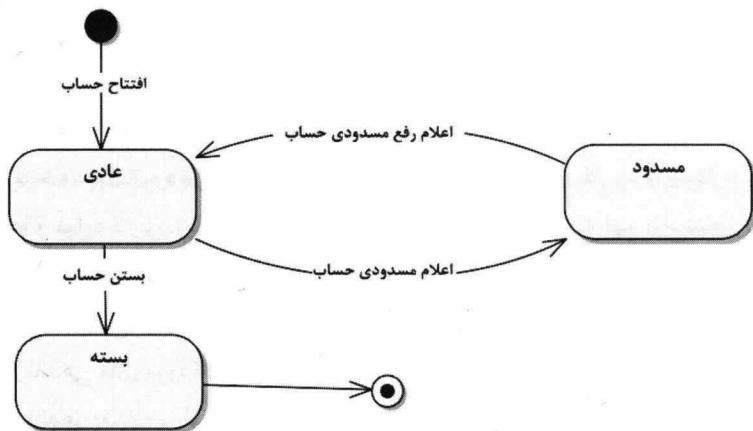
1. Background
2. Stereotype
3. Icon
4. Partition
5. Transitions

آن‌ها، باید به شیوه‌ای در راه حل ارائه شده (موارد کاربرد) پوشش داده شده باشند.

به عنوان مثال، در سیستم حساب سپرده کوتاه مدت، حساب یکی از مهم‌ترین اشیای سیستم است. وضعیت‌های این شیء در طول عمرش عبارتند از:

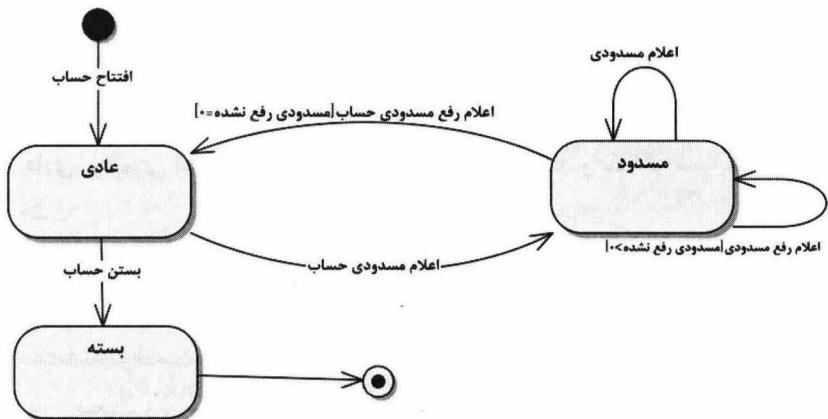
- عادی: وضعیتی است که حساب پس از افتتاح پیدا می‌کند و بیانگر شرایط عادی حساب است.
- مسدودی: وضعیتی است که در آن، حساب مسدود شده است. برداشت از حساب مسدودی، امکان‌پذیر نیست.
- بسته شده: وضعیت حساب پس از بستن است.

در نمودار وضعیت زیر، وضعیت‌ها به همراه چگونگی تبدیل آن‌ها به یکدیگر مدل شده است. در نمودار مشخص شده است که با افتتاح حساب، حساب به وضعیت عادی می‌رود. در صورت اعلام مسدود کردن حساب توسط مراجع قضایی یا دستور واحدهای بانک، وضعیت حساب به مسدودی، تغییر می‌کند. اگر حساب توسط مشتری بسته شود، وضعیت حساب به بسته شده تغییر می‌کند.



شکل ۲۱-۷: نمودار وضعیت حساب

با بررسی و تحلیل نمودار قبل، مشخص می‌گردد که تحلیل گر چندین «اعلام مسدودی» متوالی را مدنظر قرار نداده است. به عنوان مثال اگر اعلام مسدودی حساب سه بار متوالی انجام شود، سیستم نباید با اولین اعلام رفع مسدودی حساب، وضعیت آن را به عادی برگرداند. با کشف این خطای تحلیلی، نمودار وضعیت به شکل زیر تصحیح می‌گردد و به همراه آن، مشخصات مورد کاربرد اعلام مسدودی یا اعلام رفع مسدودی نیز باید تصحیح شود تا خطای مذکور رفع گردد.



شکل ۲۲-۷: نمودار وضعیت حساب

در نمودار ۲۲-۷، مشخص شده است که تنها حساب عادی می‌تواند بسته شود. در نتیجه، در مورد کاربرد «بستن حساب» باید کنترل شود که حسابی که کاربر قصد بستن آن را دارد، در وضعیت عادی باشد و مسدود نشده باشد (در قالب یک گردش جایگزین مدون می‌گردد).

۱۰- سازماندهی مدل مورد کاربرد^۱

تا قبل از این مرحله، مدل مورد کاربرد شامل مجموعه‌ای از کنشگرها، موارد کاربرد و نمودارها است. اگر تعداد کنشگرها و موارد کاربرد زیاد باشد، مدل به اندازه‌ای پیچیده می‌گردد که فهم و استنباط آن دشوار می‌شود. سازماندهی مدل مورد کاربرد، روشی است برای دسته‌بندی اجزای مدل مورد کاربرد به نحوی که با کاهش پیچیدگی، خوانایی آن را افزایش دهد.

روش‌های سازماندهی مدل مورد کاربرد عبارتند از:

- سازماندهی مبتنی بر لیست^۲
- سازماندهی مبتنی بر کنشگر
- سازماندهی مبتنی بر کارکردهای کسب و کار^۳
- سازماندهی مبتنی بر فرایندهای کسب و کار^۴
- سازماندهی مبتنی بر اولویت توسعه^۵

در این روش‌ها، کنشگرها، موارد کاربرد و نمودارها به کمک بسته دسته‌بندی می‌گردند. در ادامه، این روش‌ها معرفی می‌گردند.

1. Organizing use case model
2. List
3. Business function
4. Business process
5. Development priority

۱-۱- سازماندهی مبتنی بر لیست

این روش ساده‌ترین روش سازماندهی مدل مورد کاربرد است. در این روش، کنشگرها، موارد کاربرد و نمودارها به دنبال هم آورده می‌شوند.

شکل زیر نمونه‌ای از این روش سازماندهی را در ابزار EA نشان می‌دهد.



شکل ۲۳-۷: نمونه سازماندهی مبتنی بر لیست

۲- سازماندهی مبتنی بر کنشگر

در این روش هر کنشگر و موارد کاربرد مرتبط با آن در یک بسته جدا قرار داده می‌شوند.

موارد کاربرد مشترک بین چندین کنشگر در یک بسته جدا به نام «موارد کاربرد مشترک» قرار داده می‌شوند. به بسته‌های مرتبط با هر کنشگر، نمودار یا نمودارهایی اضافه می‌شوند که موارد کاربرد مشترک در آن‌ها نیز آورده می‌شوند.

به عنوان مثال، در سیستم حساب سپرده کوتاه مدت می‌توان مدل مورد کاربرد را به شکل زیر دسته‌بندی کرد:

- « بسته‌ای به نام «کارهای متصدی امور بانکی» شامل تمامی موارد کاربردی که توسط متصدی امور بانکی انجام می‌شوند.
- « بسته‌ای به نام «کارهای رئیس شعبه» شامل تمامی موارد کاربردی که توسط رئیس شعبه انجام می‌شوند.
- « بسته‌ای به نام «کارهای مدیریت امور شعب» شامل تمامی موارد کاربردی که توسط کارشناس مدیریت امور شعب انجام می‌شوند.
- « بسته‌ای به نام «کارهای راهبر سیستم» شامل تمامی موارد کاربردی که توسط راهبر سیستم انجام می‌شوند.
- « بسته‌ای به نام «کارهای مشترک» که در آن موارد کاربرد مشترک کنشگرها مانند «ورود به سیستم» قرار داده می‌شوند.

این روش برای نمایش کارهای هر کنشگر در سیستم، مناسب است.



شکل ۲۴-۷: نمونه سازماندهی مبتنی بر کنشگر

۳-۱۰- سازماندهی مبتنی بر کارکردهای کسب و کار

در این روش، کنشگرها و موارد کاربرد مربوط به یک کارکرد کسب و کار، در یک بسته قرار می‌گیرند. به عنوان مثال در سیستم حساب سپرده کوتاه مدت، به کارگیری این روش منجر به شناسایی بسته‌های زیر می‌گردد:

- » بسته‌ای به نام «شعبه» که در آن کنشگرهای رئیس شعبه و منتصدی امور بانکی به همراه تمامی موارد کاربردی که در شعبه انجام می‌شود، قرار داده شده‌اند.
 - » بسته‌ای به نام «مدیریت امور شب» که در آن کنشگرهای کارشناس امور شب به همراه تمامی موارد کاربردی که در مدیریت امور شب انجام می‌شود، قرار داده می‌شوند.
- این روش برای ارزیابی جامعیت راه حل برای گروههای کاری مناسب است. به علاوه نتایج این روش، ورودی بسیار مناسبی برای تعیین معماری سیستم است.



شکل ۲۵-۷: نمونه سازماندهی مبتنی بر کارکردهای کسب و کار

۱۰-۴- سازماندهی مبتنی بر فرایندهای کسب و کار

در این روش، معیار تعیین بسته‌ها، فرایندهای کسب و کار هستند. ابتدا برای هر فرایند یا مجموعه‌ای از فرایندهای مرتبط، بسته‌ای ایجاد می‌گردد و سپس، کنشگرهای، موارد کاربرد و نمودارهای مرتبط در آن قرار می‌گیرند.

به عنوان مثال، فرایند خرید و فرایند فروش، فرایندهای مهم و بزرگی در سازمان هستند. با استفاده از روش بالا، دو بسته به نام فرایندهای مذکور ایجاد می‌گردد و اجزای مرتبط در آن قرار می‌گیرند. این روش برای تحلیل‌هایی که مبتنی بر فرایند انجام می‌شوند و نیز بررسی جامعیت فرایندهای کسب و کار مفید است.

۱۰-۵- سازماندهی مبتنی بر اولویت توسعه

در این روش، موارد کاربردی که دارای اولویت یکسانی برای پیاده‌سازی هستند، در یک بسته قرار می‌گیرند. به عنوان مثال، تمامی موارد کاربردی که قرار است در نسخه شماره یک نرمافزار، پیاده‌سازی شوند، در بسته‌ای به نام «نسخه ۱» و موارد کاربردی که در نسخه دوم، پیاده‌سازی خواهند گردید، در بسته‌ای به نام «نسخه ۲» قرار می‌گیرند.

۱۱- توضیحات تکمیلی

توصیه‌های پیشنهادی مانحصراً تجارت اجرای مراحل پیشنهادی در تیم‌های تحلیل مختلف است. ممکن است، تیم‌ها قواعد و توصیه‌های دیگری را نیز تبیین و اجرا کنند.

هر چند تدوین مدل و مشخصات مورد کاربرد، ماهیتی تدریجی دارد، اما مراحل پیشنهادی در این فصل با هدف آموختن گام به گام خوانندگان ارائه شده است. پس از اجرای آن در چند پروژه، تحلیل‌گران می‌توانند به شیوه دلخواه خود، مبادرت به تدوین مدل و مشخصات مورد کاربرد نمایند. با اجرای هر مرحله، امکان تغییر سند واژه‌نامه، سند چشم‌انداز، مدل تحلیل یا نمونه اولیه^۱ وجود دارد.

۱۲- نکات کلیدی

- تدوین مدل و مشخصات مورد کاربرد به صورت تدریجی انجام می‌شود.
- قواعد پوشش‌داده شده در مورد کاربرد، در مشخصات آن مستند می‌گرددند.
- تعیین روابط بین موارد کاربرد و بسته‌بندی مدل برای سازماندهی مدل مورد کاربرد انجام می‌شود.

از سایر نمودارها مانند نمودار فعالیت و نمودار وضعیت می‌توان در مدل مورد کاربرد استفاده کرد.

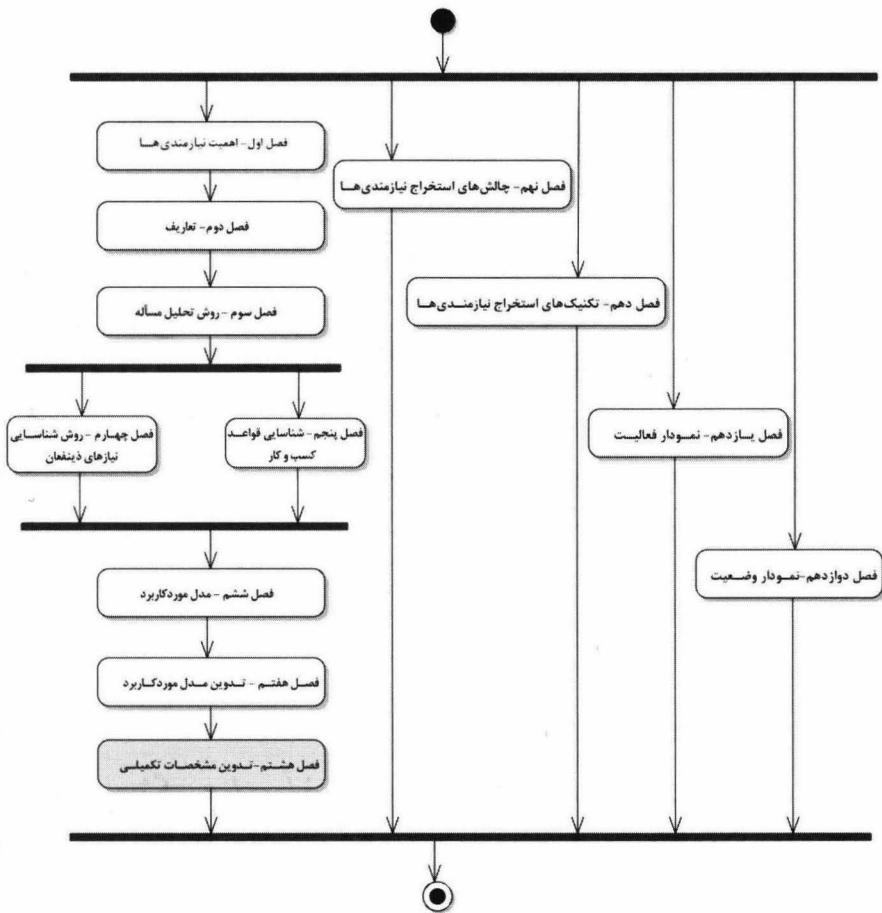
۱۳- مراجع

1. Cockburn, Alistair. Wrriten Effective Use Cases. Addison Wesley
2. Davis, Alan M. Software Requirements: Objects, Functions, and States. Prentice-Hall, 1993
3. Gunnar, Övergaard, Karin Palmkvist. Use Cases Patterns and Blueprints. Addison Wesley, 2004
4. James, Heumann. "Tips for writing good use cases." 2008
5. Leffingwell, Dean, Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
6. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
7. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>

فصل هشتم - تدوین مشخصات تکمیلی

افزودن ویژگی‌ها در آخرین دقایق به دلیل فشار رقابتی یا علاقه توسعه‌دهندگان یا تمایل و دستور مدیریت، بیش از هر عامل دیگری باعث افزایش خطاهای نرم‌افزار می‌شود.

جان رابینز نویسنده کتاب *Debugging Applications*



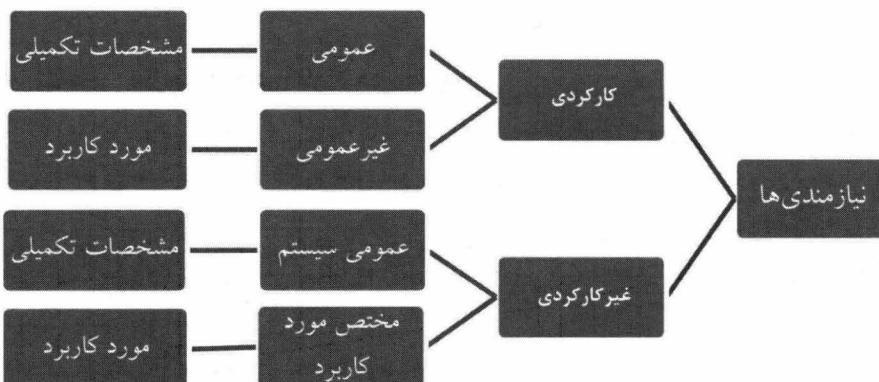
۱- مقدمه

در این فصل، مجموعه‌های از نیازمندی‌های سیستم که با نام مشخصات تکمیلی شناخته می‌شوند، معرفی و تشریح می‌گردد. ابتدا به معرفی جایگاه این گونه نیازمندی‌ها پرداخته می‌شود و در ادامه مراحل شناسایی آن‌ها ارائه خواهد شد. در پایان نیز عنوانین سند «مشخصات تکمیلی» که محل تدوین نیازمندی‌های مذکور است، توضیح داده شده است.

۲- جایگاه مشخصات تکمیلی

در فصل‌های قبل گفته شد که نیازمندی‌های کارکردی سیستم به شکل موارد کاربرد شناسایی و مدون می‌گردد. نیازمندی‌های غیرکارکردی مختص به هر مورد کاربرد نیز در قسمت «نیازمندی‌های خاص» آن تدوین می‌گردد.

برخی از نیازمندی‌های نرم‌افزاری به شکل موارد کاربرد قابل بیان نیستند. نیازمندی «برنامه باید بر روی سیستم عامل ویندوز ایکس‌پی کار کند» نمونه‌ای از این گونه نیازمندی‌هاست. از آنجا که موارد کاربرد اکثر نیازمندی‌های سیستم را در خود جای داده‌اند، این گونه نیازمندی‌ها تکمیل‌کننده موارد کاربرد هستند. از این رو آن‌ها را «تکمیلی» می‌نامیم و در سندي به نام «مشخصات تکمیلی» تدوین می‌کنیم. چگونگی تدوین نیازمندی‌ها در شکل زیرنشان داده شده است.



شکل ۸-۱: چگونگی تدوین نیازمندی‌ها در مستندات

سند «مشخصات تکمیلی» خصایص کیفی، قیدهای حاکم بر سیستم نرم‌افزاری و نیازمندی‌های کارکردی عمومی سیستم را شامل می‌شود. به عبارت دیگر، این سند شامل نیازمندی‌هایی از سیستم است که به سادگی در مشخصات موارد کاربرد قابل تدوین نیستند. اهداف تهیه سند مشخصات تکمیلی عبارتند از:

- توصیف خصایص کیفی سیستم، تدوین نیازمندی‌های غیرکارکردی، استانداردها و قیدهایی که در طراحی باید در نظر گرفته شوند.

- استخراج نیازمندی‌های کارکردی که نمی‌توان آن‌ها را به شکل موارد کاربرد تدوین نمود.
- هر چند در بسیاری از سیستم‌ها اکثر نیازمندی‌های سیستم به شکل موارد کاربرد شناسایی می‌گردد، اما این موضوع دلیل بر کم‌اهمیت بودن نیازمندی‌های تکمیلی سیستم نیست. تا جایی که حتی با وجود درستی کارکرد موارد کاربرد در سیستم، عدم پوشش مشخصات تکمیلی می‌تواند منجر به شکست پروژه گردد. به عنوان مثال در نظر نگرفتن نیازمندی «استفاده از قرارداد TLS¹ در ارتباطات شبکه‌ای» در طراحی نرم‌افزار منجر به عدم پذیرش سیستم توسط کاربر خواهد شد. با این وجود، بر خلاف سیستم‌های معمولی، در برخی از سیستم‌ها، «مشخصات تکمیلی» بخش بزرگی از نیازمندی‌های سیستم را نسبت به موارد کاربرد تشکیل می‌دهد.

۳- شناسایی مشخصات تکمیلی

در این قسمت، گام‌های شناسایی مشخصات تکمیلی تشریح شده است. این گام‌ها عبارتند از:

- شناسایی نیازمندی‌های کارکردی عمومی
- شناسایی مشخصات کفی سیستم
- شناسایی قیدهای طراحی
- شناسایی نیازمندی‌های الزامی
- شناسایی نیازمندی‌های مستندسازی
- شناسایی سایر نیازمندی‌ها

۱-۳- شناسایی نیازمندی‌های کارکردی عمومی

- هر چند مورد کاربرد یکی از مهمترین ابزارهای استخراج نیازمندی‌های سیستم است و نیازمندی‌های کارکردی به خوبی به شکل موارد کاربرد قابل تدوین هستند، اما برخی از نیازمندی‌های کارکردی به راحتی در قالب مورد کاربرد قابل بیان نیستند. بسیاری از سیستم‌ها علی‌رغم دارا بودن مجموعه زیادی از نیازمندی‌های کارکردی، دارای تعامل بسیار کمی با کاربر یا سایر کنسلگرهای هستند. سیستم‌های زیر، نمونه‌ای از این دست سیستم‌ها هستند.

- سیستم‌هایی که الگوریتم یا محاسبات خاصی را انجام می‌دهند. به عنوان مثال سیستم مستقل محاسبه سود که در تاریخ مشخصی از هر ماه شروع به کار کرده، سود حساب‌های سپرده کوتاه مدت را محاسبه کرده، به حساب مشتریان واریز می‌کند. این سیستم هیچ‌گونه تعاملی با کاربر ندارد اما دارای مجموعه زیادی از نیازمندی‌های کارکردی است که بیان‌گر روش محاسبه سود در شرایط مختلف است.

- بسیاری از برنامه‌ها در پشت صحنه کار می‌کنند و تعامل بسیار کمی با کاربر و سایر کنşگران دارند. به عنوان مثال برنامه ضدوبروز که عمدۀ نیازمندی‌های کارکردی آن به روش شناسایی ویروس‌ها و کترول برنامه‌ها و فایل‌ها مرتبط است.
- برنامه‌هایی که مانند کامپایلر عمل می‌نمایند: داده‌های ورودی را دریافت و پس از تجزیه و تحلیل آن‌ها، خروجی‌هایی تولید می‌کنند. برنامه مبدل داده‌های مبتنی بر یک صفحه کد^۱ به صفحه کد دیگر، نمونه‌ای از این برنامه‌ها است.
- در این گونه سیستم‌ها «مشخصات تکمیلی» نقش مهم‌تری در فرآیند نیازمندی‌ها بر عهده دارد. در این سیستم‌ها، بسیاری از نیازمندی‌های کارکردی در «مشخصات تکمیلی» ثبت می‌شود و مدل مورد کاربرد نیازمندی‌های کمتری را شامل می‌شود. در این شرایط، نیازمندی‌های کارکردی به روش «توصیفی» (متنی با جملات ساده) و در قسمت «نیازمندی‌های کارکردی» سنده «مشخصات تکمیلی» بیان می‌گردد.
- از جمله نیازمندی‌های عمومی که معمولاً در قالب موارد کاربرد تدوین نمی‌گردد می‌توان به امکانات گزارش‌گیری، ممیزی^۲، چاپ، امنیت، مجوز استفاده^۳ و احراز هویت^۴ اشاره کرد.

۲-۳- شناسایی خصایص کیفی سیستم

- نیازمندی‌های غیرکارکردی شامل مشخصات کیفی و قیدها هستند. در این گام از تدوین مشخصات تکمیلی سیستم، مشخصات کیفی شناسایی می‌گردد.
- مشخصات کیفی سیستم، ویژگی‌ها و خصوصیاتی از سیستم‌اند که مورد توجه ذینفعان هستند و از این رو بر میزان رضایتشان از سیستم تأثیرگذارند. مشخصات کیفی شامل URPS+^۵ FURPS+ هستند که عبارتند از خوش‌کاری، قابلیت اطمینان، کارایی و قابلیت پشتیبانی. در ادامه راهنمایی‌ها و مثال‌هایی برای شناسایی و تدوین این نیازمندی‌ها ارائه می‌گردد.

۲-۳-۱- خوش‌کاری

- امروزه سادگی استفاده یکی از معیارهای موقیت اقتصادی و رضایت کاربران در توسعه محصولات نرم‌افزاری است. از آنجا که سادگی استفاده از دید افراد مختلف، متفاوت است، تعیین آن‌ها، برای تیم‌های توسعه چالشی بزرگ است.

- راه ساده‌ای برای تعیین این گونه نیازمندی‌ها وجود ندارد و مناسب‌ترین راه، به کارگیری مجموعه‌ای از مصادیق و مثال‌ها است. برخی از این موارد عبارتند از:
- حداقل زمان آموزش یک کاربر برای یادگیری انجام یک کار ساده و انجام کارهای روزانه را به تفکیک کاربران نوآموز، معمولی و حرفه‌ای مشخص نمایید.

1. Code page

2. Audit

3. License

4. Authentication

5. Functionality, Usability, Reliability, Performance, Supportability, + (others)

- زمان لازم برای انجام کارها یا تراکنش‌ها توسط کاربر را مشخص کنید.
- استانداردهای عمومی رابط کاربر (مانند استانداردهای رابط کاربر مایکروسافت^۱) را در صورت الزامی بودن مشخص نمایید.
- سادگی استفاده سیستم جدید را با سیستم‌های معتبر مشابهی که کاربران با آن‌ها آشنا هستند یا توسط کاربران مشابه استفاده می‌گردد، مقایسه کنید. به عنوان مثال «رابط کاربر سیستم سپرده شبیه به رابط کاربر سیستم بانکی آی‌بی‌ام باشد».
- نیازمندی‌های مرتبط با راهنمای کاربران و مشخصات آن، آموزشگرها^۲، راهنمای ابزارها^۳ و کتابچه راهنمای^۴ را مشخص نمایید.

تلاش‌های زیادی برای تثیت مفهوم و مصاديق خوش‌کاری انجام شده است. یکی از این تلاش‌ها منجر به منشوری ده ماده‌ای به شکل زیر شده است: (Karat, 1998)

۱. همیشه حق با کاربر است. اگر مشکلی در استفاده از سیستم وجود دارد، این سیستم است که مشکل دارد نه کاربر.
۲. حق کاربر است که بتواند بدون وقوع اثرات منفی، سیستم سخت‌افزاری و نرم‌افزاری را به آسانی نصب یا حذف نماید.
۳. حق کاربر است که سیستم همان‌طوری که تعهد شده است، کار کند.
۴. حق کاربر است که برای شناخت و استفاده از سیستم از دستورالعمل‌های ساده^۵ (شامل کتابچه راهنمای لحظه‌ای و پیغام‌های خطای خطا) استفاده نماید.
۵. حق کاربر است که سیستم کارهایش را کنترل کرده، به خواسته‌های وی پاسخ دهد.
۶. ارائه اطلاعات واضح، قابل فهم و دقیق از کار فعلی و میزان پیشرفت آن توسط سیستم، حق کاربر است.
۷. اطلاع دقیق از نیازمندی‌های سیستم برای استفاده موفق از آن، حق کاربر است.
۸. اطلاع از حد و مرز قابلیت‌های سیستم از حقوق کاربر است.
۹. در صورت نیاز، ارتباط با ارائه‌دهنده فناوری و دریافت پاسخ منطقی و راه‌گشا از وی، از حقوق کاربر است.
۱۰. کاربر باید حاکم بر فناوری نرم‌افزار و ساخت‌افزار باشد و نه برعکس. محصولات باید به صورت طبیعی و شهودی قابل استفاده باشند.

1. Microsoft GUI Standard
 2. Wizards
 3. Tool tips
 4. User manual
 5. Easy-to-use

منشور مذکور می‌تواند مرجعی برای شناسایی معیارهای مؤثر بر خوشکاری سیستم باشد.

۲-۲-۳-قابلیت اطمینان

هیچکس خطأ، نقص و از دست دادن اطلاعات در سیستم را دوست ندارد و از آنجا که معمولاً اثری از وجود خطأ، نقص یا از دست دادن اطلاعات در نیازمندی‌ها وجود ندارد و فرض بر این است که سیستم درست کار خواهد کرد، کاربر نیز انتظار وجود آن‌ها را در سیستم ندارد. امروزه در دنیای کامپیوتر، حتی خوشبینانه‌ترین کاربران نیز به وجود اشتباه و خطأ در نرم‌افزار اطمینان دارند. بنابراین باید نیازمندی‌های مرتبط با رفتار قابل قبول سیستم از دید کاربران توصیف گرددند. در ادامه برخی از این گونه نیازمندی‌ها تشریح شده‌اند.

○ دسترس پذیری^۱

سیستم باید در زمان‌های مشخصی در دسترس کاربران باشد. در حالت ایده‌آل، سیستم باید بدون وقفه در دسترس باشد(۲۴ ساعت شبانه روز، ۳۶۵ روز سال). اما بسته به نوع سیستم، این بازه ایده‌آل ممکن است تغییر کند. به عنوان مثال، این معیار برای سیستم بانک اینترنتی (۳۶۵ روز، ۲۴ ساعت) با سیستم حسابداری یک شرکت(روزهای کاری سال، ۸ ساعت کاری) متفاوت است. علاوه بر تعیین بازه مورد استفاده، باید مشخص گردد که کدام امکانات سیستم در هر یک از بازه‌ها مورد نیاز است.

○ میانگین زمانی بین خرابی‌ها یا MTBF^۲

این معیار بیان‌کننده میانگین زمانی بین دو خرابی و از کارافتادن سیستم است و معمولاً به واحد ساعت یا بر حسب روز، ماه یا سال بیان می‌گردد. دشواری تعیین این معیار به توافق بر سر تعریف خرابی بر می‌گردد.

○ میانگین زمان تعمیر یا MTTR^۳

این معیار بیان‌گر مدت زمان مجاز برای در دسترس نبودن سیستم در هنگام تعمیر، پس از وقوع خرابی است. دشواری تعیین این نیازمندی به نامشخص بودن زمان حضور تیم در محل پس از وقوع خرابی، زمان لازم برای شناسایی عامل خرابی و زمان لازم برای رفع آن بر می‌گردد. به طور معمول، تیم‌های توسعه تنها مدت زمان حضور در محل پس از وقوع خرابی را تعهد می‌کنند.

○ دقت^۴

این نیازمندی میزان دقت در خروجی‌های سیستم مانند خروجی‌های عددی را مشخص می‌کند.

○ حداکثر تعداد خطأ یا نرخ نقص‌ها^۵

1. Availability

2. Mean Time Between Failures

3. Mean Time To Repair

4. Accuracy

5. Maximum bugs, or defect rate

این نیازمندی بیانگر تعداد خطاهای نقص‌های قابل قبول است که معمولاً با واحد «تعداد خطای هزار خط کد^۱» یا خطای در نقاط کارکرد^۲ بیان می‌گردد.

○ تعداد هر نوع از خطای

این نیازمندی بیانگر تعداد خطاهای قابل قبول از هر نوع خطای است که در قالب دسته‌هایی مانند «کم اهمیت»، «مهم» و «بحارانی» بیان می‌گردد. در اینجا تعریف دقیق هر دسته بسیار حائز اهمیت است به عنوان مثال خطای بحرانی خطایی است که در آن داده‌های سیستم از دست بروید یا نتوان از بخشی از سیستم استفاده نمود.

۳-۲-۳- کارایی

نیازمندی‌های کارایی شامل دسته‌های زیر می‌گردد:

○ زمان پاسخ^۳

این نیازمندی بیانگر میانگین یا بیشینه مدت زمان لازم برای انجام یک کار یا تراکنش است. به عنوان مثال کلیه گزارش‌های سیستم باید در کمتر از پنج ثانیه به کاربر نشان داده شوند.

○ توان عملیاتی

این نیازمندی، حداقل ظرفیت سیستم را برای انجام یک کار بیان می‌کند که ممکن است به صورت تعداد تراکنش در هر ثانیه بیان شود.

○ ظرفیت^۴

در این گونه نیازمندی، حجم عملیاتی را که سیستم قادر به پشتیبانی از آن است، مشخص می‌شود. برای مثال می‌توان به تعداد کاربران همزمان در انجام کارهای سیستم اشاره نمود.

○ مصرف منابع

در این نوع نیازمندی، میزان مصرف منابع مانند دیسک، حافظه و خطوط ارتباطی توسط سیستم مشخص می‌گردد.

۴-۲-۴- قابلیت پشتیبانی

قابلیت پشتیبانی به معنای توانایی تعمیر یا توسعه سیستم به ساده‌ترین روش ممکن است. در برخی از سیستم‌ها تغییرات قابل پیش‌بینی هستند. این گونه نیازمندی بیانگر زمان پاسخ به تغییرات ساده، متوسط و پیچیده است که تیم پشتیبانی به آن متوجه شده است.

1. Bug/KLOC(thousands of lines of code)

2. Bug per function-point

3. Response time

4. Capacity

برای مثال در سیستم حساب سپرده کوتاه مدت، فرمول محاسبه سود علی الحساب و قطعی در هر سال تغییر می‌کند. در نیازمندی‌ها قید می‌شود که این تغییر باید حداقل تا قبل از شروع تاریخ اجرای دستورالعمل محاسبه سود در سیستم اعمال گردد. بدیهی است اعمال این تغییر به سادگی ممکن نیست و تیم توسعه به طراحی روش منعطفی در سیستم نیاز دارد.

از جمله مواردی که در این گونه نیازمندی‌ها بیان می‌شود، می‌توان به موارد زیر اشاره نمود:

- استانداردهای کدنویسی^۱
- قراردادهای نام‌گذاری^۲
- کتابخانه‌های^۳ مورد استفاده

۳-۳- شناسایی قیدهای طراحی

قیدهای طراحی بیانگر گزینه‌های قابل انتخاب در طراحی یا محدودیت‌های طراحی سیستم هستند.

قیدهای طراحی ناشی از «محدود کردن گزینه‌های انتخابی برای طراحی»، «الزامات اعمال شده بر فرآیند توسعه» یا «آیین‌نامه‌ها و استانداردهای تحمیلی» است.

برای طراحی اکثر نیازمندی‌ها، بیش از یک گزینه وجود دارد. همواره توصیه می‌گردد که دست طراحان در انتخاب گزینه‌ها باز گذاشته شود، چرا که آنان در بررسی و انتخاب راه حل‌های مناسب طراحی با توجه به شرایط فنی و هزینه‌ای، توانایی و صلاحیت لازم را دارند. با این وجود، اگر انتخاب گزینه‌های طراحی مانند نوع بانک اطلاعاتی از طراحان سلب گردد، این کار منجر به شناسایی قیدی در طراحی می‌گردد که از درجه آزادی طراحان خواهد کاست.

نوع دیگری از قیدهای طراحی ناشی از اعمال نیازمندی‌هایی بر فرآیند توسعه سیستم است. به نمونه‌های زیر توجه کنید:

- سازگاری با سیستم‌های فعلی: سیستم باید بر روی هر دو سکوی^۴ جدید و قدیم مشتری قابل اجرا باشد
- استانداردهای برنامه‌های کاربردی: استفاده از مؤلفه امنیت و حقوق دسترسی کاربران موجود در سیستم‌های فعلی
- استانداردها و نمونه‌های موفق مشتری: سازگاری با بانک اطلاعاتی قبلی مشتری آیین‌نامه‌ها و استانداردهای حاکم بر حوزه کسب و کار سیستم یکی دیگر از منابع مهم برای شناسایی قیدهای طراحی هستند. این محدودیت‌ها ممکن است نه تنها بر طراحی محصول، بلکه بر فرآیند توسعه آن نیز اعمال گردد. با توجه به حجمی و طویل بودن قوانین و آیین‌نامه‌ها، به جای ذکر مجدد آنها در مستندات، بهتر است به آنها ارجاع داد. در تدوین قیدهای طراحی، بهتر است مأخذ و منبع هر یک مشخص شود تا علاوه بر استفاده

منابع در بازنگری، در آینده نیز بتوان از آن‌ها استفاده نمود. به علاوه بهتر است دلیل وجودی هر قید در حداکثر دو جمله مشخص گردد. این کار باعث یادآوری انگیزه وجود قیدها در آینده می‌گردد.

۳-۴- شناسایی نیازمندی‌های الزامی^۱

در این مرحله الزامات مرتبط با استانداردها شناسایی و تدوین می‌گردد. نیازمندی‌های الزامی ممکن است در قالب سایر نیازمندی‌ها (کارکردی، غیرکارکردی یا قیدها) بیان گردد. آن‌چه اهمیت دارد، شناسایی، تدوین و کسب توافق درباره آنها است. جدا از محل تدوین (در قالب نیازمندی‌های کارکردی، غیرکارکردی یا قیدها)، لازم است آن‌ها را به صورت دقیق مشخص کرد.

در صورتی که تعداد این نیازمندی‌ها زیاد باشد، می‌توان آن‌ها را در بخش‌های جداگانه‌ای تدوین نمود.

دسته‌بندی زیر نمونه‌ای از آن است:

- نیازمندی‌های مجوز استفاده
- حق تأليف، ضمان و علامت تجاری
- استانداردهای کاربردی

۳-۵- شناسایی نیازمندی‌های مستندسازی

در این مرحله، نیازمندی‌های مرتبط با مستندات سیستم مدون می‌گردد. این نیازمندی‌ها شامل نیازمندی‌های مرتبط با راهنمای لحظه‌ای، راهنمای نصب و راهاندازی، کتابچه راهنمای کاربران و مطالب آموزشی است. همانند نیازمندی‌های الزامی، نیازمندی‌های مستندسازی نیز ممکن است در قالب سایر نیازمندی‌ها مدون گردد.

۳-۶- شناسایی سایر نیازمندی‌ها

با وجود لیست کاملی که تاکنون برای نیازمندی‌ها بیان شد، ممکن است نیازمندی‌های دیگری نیز وجود داشته باشند. برخی از منابع این نیازمندی‌ها عبارتند از:

- خروجی‌های فیزیکی تولید شده توسط سیستم مانند سی‌دی
- پیکربندی و آماده‌سازی سیستم برای استقرار
- نیازمندی‌های مرتبط با پشتیبانی یا آموزش
- نیازمندی‌های مرتبط با جهانی‌سازی یا محلی‌سازی^۲

۴- قالب مستند مشخصات تکمیلی

نیازمندی‌های مربوط به مشخصات تکمیلی در سندي به نام «مشخصات تکمیلی» تدوین می‌گردد. در ادامه، قالب این سندي به همراه توضیحات تکمیلی آن ارائه شده است.

1. Compliance

2. Internationalization and localization requirements

-۱ مقدمه

در این قسمت، مقدمه‌ای بر مستند نوشته می‌شود. مقدمه در قالب قسمت‌های هدف، واژگان، محدوده، مراجع و مرور تقسیم‌بندی می‌گردد.

-۱-۱ هدف

هدف از ایجاد مستند، شرح داده می‌شود.

-۱-۲ محدوده

در این قسمت، خلاصه‌ای از محدوده مستند بیان می‌گردد.

-۱-۳ واژگان

در این قسمت، واژگان، سرنام‌ها و اصطلاحات توضیح داده می‌شود. معمولاً این قسمت به سند واژه‌نامه ارجاع می‌شود.

-۱-۴ مراجع

در این قسمت، لیستِ مستندات ارجاع شده در این مستند آورده می‌شود.

-۱-۵ مرور

در این قسمت، بخش‌های سند و چگونگی بخش‌بندی آن‌ها شرح داده می‌شود.

-۲ نیازمندی‌های وظیفه‌مندی

در این قسمت، آن دسته از نیازمندی‌های وظیفه‌مندی که در قالب مورد کاربرد نمی‌گنجند و باید به شیوه توصیفی بیان گردد، نوشته می‌شود.

-۳ خوش‌کاری

نیازمندی‌های تأثیرگذار بر خوش‌کاری در این بخش آورده می‌شوند.

-۴ قابلیت اطمینان

نیازمندی مرتبط با قابلیت اطمینان در این بخش مدون می‌شوند.

-۵ کارایی

خصوصیات مشترک مرتبط با کارایی بین تمامی موارد کاربرد، در این قسمت مدون می‌شوند.

-۶ قابلیت پشتیبانی

نیازمندی‌های قابلیت پشتیبانی و نگهداری سیستم در این قسمت ارائه می‌شوند.

-۷ قیدهای طراحی

قیدهای طراحی سیستم و فرایند در این بخش ذکر می‌شوند.

-۸ نیازمندی‌های مستندسازی

نیازمندی‌های مرتبط با مستندات کاربران و مدیران سیستم در این قسمت ذکر می‌شوند.

-۹ مؤلفه‌های خریداری شده

لیست مؤلفه‌هایی که باید برای توسعه و راهاندازی سیستم خریداری شوند، مجوز استفاده، محدودیت‌های

کاربردی و رابطه‌های استاندارد آن‌ها در این قسمت مستند می‌شوند.

- ۱۰ - رابطه‌ها

در این قسمت، رابطه‌ای که سیستم باید آن‌ها را پشتیبانی کند، مدون می‌گردد.

- ۱۱ - رابطه‌های کاربر

- ۱۲ - رابطه‌ای سخت‌افزاری

- ۱۳ - رابطه‌ای نرم‌افزاری

- ۱۴ - رابطه‌ای ارتباطی

- ۱۵ - نیازمندی‌های مجوز استفاده

نیازمندی‌های مرتبط با امنیت، دسترس‌پذیری سیستم و قوانین استفاده را تشریح می‌کند.

- ۱۶ - قانون، حق تأییف و سایر موارد

قوانین و نکات مرتبط با حق تأییف در این قسمت بیان می‌شوند.

- ۱۷ - استانداردهای کاربردی

تمام استانداردهایی که باید در سیستم رعایت شوند، در این قسمت ارجاع داده می‌شوند.

۵- نکات کلیدی

- برخی از نیازمندی‌های نرم‌افزاری به شکل موارد کاربرد قابل بیان نیستند.
 - سند «مشخصات تکمیلی» خصایص کیفی، قیدهای حاکم بر سیستم نرم‌افزاری و نیازمندی‌های کارکردی عمومی سیستم را شامل می‌شود.
 - در سند «مشخصات تکمیلی»، نیازمندی‌های کارکردی به روش «توصیفی» (منتی با جملات ساده) و در قسمت «نیازمندی‌های کارکردی» بیان می‌گردد.
 - نیازمندی‌های عمومی مانند امکانات گزارش‌گیری، ممیزی، چاپ، امنیت، مجوز استفاده و احرار هویت معمولاً در قالب موارد کاربرد تدوین نمی‌گردد.
 - خصایص کیفی سیستم مانند خوش کاری، قابلیت اطمینان و کارایی در سند نیازمندی‌های تکمیلی مستندسازی می‌شوند.
 - قیدهای طراحی در سند نیازمندی‌های تکمیلی مستندسازی می‌شوند.
- آیین‌نامه‌ها و استانداردهای حاکم بر حوزه کسب و کار سیستم، یکی از منابع مهم برای شناسایی قیدهای طراحی است.

۶- مراجع

1. Heumann, James. "Tips for writing good use cases." 2008
2. Leffingwell, Dean, Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
3. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
4. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>
5. Wei Ng, Pan. "Understanding types of use cases and artifacts." 2003



بخش پنجم: موضوعات تکمیلی

در این بخش دو موضوع تکمیلی و مرتبط با نیازمندی‌های نرم‌افزاری ارائه شده است. موضوع اول، چالش‌های مهم در استخراج نیازمندی‌ها و موضوع دوم، تکنیک‌های استخراج نیامندی‌ها است.

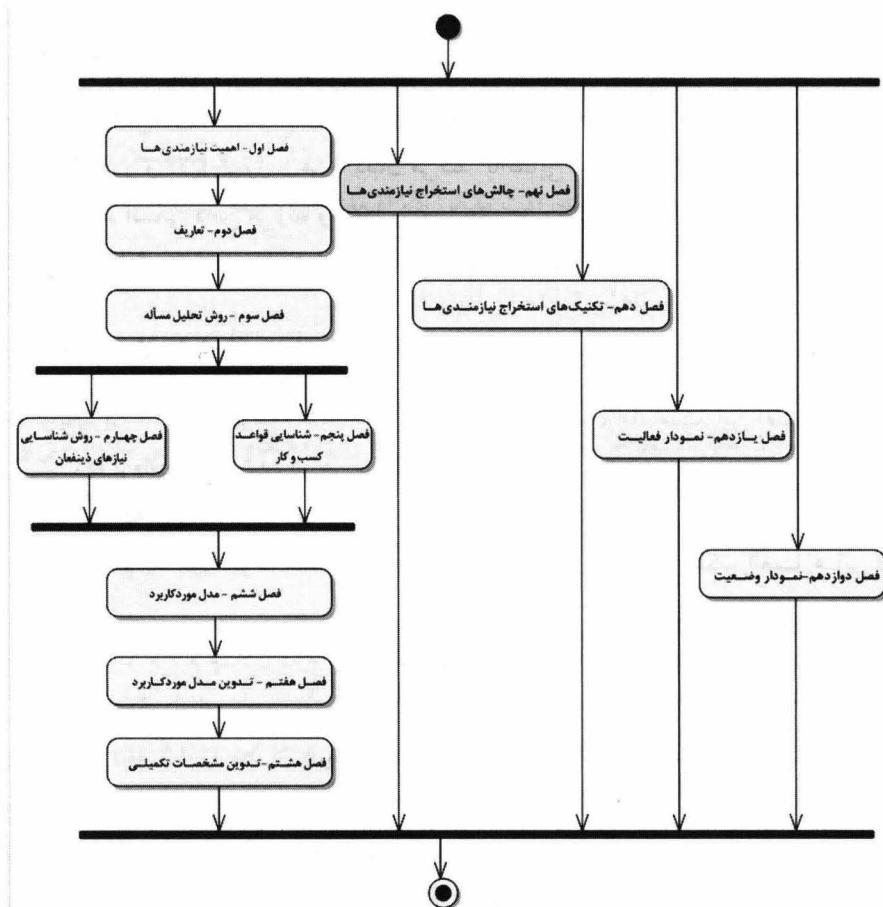
فصل نهم – چالش‌های استخراج نیازمندی‌ها

فصل دهم – تکنیک‌های استخراج نیازمندی‌ها

فصل نهم - چالش‌های اسخراج نیازمندی‌ها

ای بسا هندو و ترک هم زبان
همدلی از هم زبانی خود دیگر است
پس زبان محرومی خود دیگر است

مولانا



۱- مقدمه

در این فصل چالش‌های استخراج نیازمندی‌ها معرفی می‌گردد. این فصل ایجاد ساختاری از چالش‌های مذکور و دلایل بروز آن‌ها را دنبال می‌کند. ناآشنایی تیم توسعه با چالش‌های یادشده به مشکلات روابط انسانی دامن می‌زند و آن‌ها را بغرنج‌تر می‌کند. چالش‌های پیش‌روی استخراج نیازمندی‌ها و توسعه نرم‌افزار بسیار است. در اینجا تنها به برخی از آن‌ها اشاره شده است.

۲- آفت «بله، اما»^۱

یکی از رایج‌ترین مشکلات توسعه سیستم‌های نرم‌افزاری آفت «بله، اما» است. این آفت با عکس‌العمل کاربر نسبت به نرم‌افزار تولیدشده در هر مرحله مرتبط است.

معمولًاً مواجهه کاربران با سیستم یا بخشی از آن با واکنش‌های زیر همراه است:

- (سیستم چه قدر خوب شده، چه کار شسته‌رفته و تر و تمیزی؟) یا عکس‌العمل‌هایی از این قبیل.
- «بله، اما ام م م، الان دارم فکرمی‌کنم که اگر ، درمورد این که ...؟ بهتر نبود اگر...؟! چه می‌شد اگر ...؟!»

ریشه آفت «بله، اما» را باید در طبیعت ذهنی و ناملموس نرم‌افزار دانست. تیم توسعه می‌بایست قبل از ارائه محصول نهایی با ارائه هر چیزی که بتواند محصول را ملموس و قابل ارزیابی نماید، زمینه شناخت بهتر محصول توسط کاربر را فراهم سازد.

در طی مراحل قبلی، بررسی‌ها و مذاکرات تنها از روی مستندات و توضیحات شفاهی صورت گرفته است. تا قبل از اولین ارائه سیستم، کاربر تنها تصویری ذهنی از سیستم-بُر اساس خوانده‌ها و شنیده‌ها- دارد. دور از ذهن نخواهد بود که وی برداشت نادرستی از مطالب و مستندات داشته باشد و تصور خود را بر آن اساس شکل داده باشد. حال فرصتی دست داده است تا سیستم را از نزدیک مشاهده و حتی استفاده کند. تجربه نشان داده که تصور ذهنی وی با محصول نهایی نه تنها مطابقت ندارد، بلکه در تناقض است.

فرایند توسعه نرم‌افزار را با ساخت ابزاری مکانیکی مقایسه کنید که فناوری آن پیش از تولد نرم‌افزار برای ده‌ها سال وجود داشته است. سیستم مکانیکی دارای یک فرایند تولید مشخص و استاندارد است که شامل ساخت نمونه‌های آزمایشی، نقشه‌ها، مدل‌ها و ساخت تدریجی است که امکان مشاهده و کار کردن با محصول نیم‌ساخته و نهایی را برای کاربر فراهم می‌کند. در مقابل در تولید نرم‌افزار به اشتباه انتظار می‌رود تا بدون فراهم‌سازی امکان مشاهده و اظهار نظر پیش از اتمام

1. "Yes, But" syndrome

ساخت، محصول انتظارات کاربر را برآورده سازد. برخلاف ابزار مکانیکی، نرم‌افزار مقوله‌ای ذهنی است و این تفاوت به تشديد مشکل مذکور می‌انجامد.

ناآشنایی کاربر با سیستم‌های نرم‌افزاری تنها عامل بروز آفت نیست بلکه آفت مذکور زاییده خصایص پشتری است. چه بسیار پرورزه‌های توسعه سیستم که کاربرانش دارای تحصیلات دانشگاهی و تجارب توسعه نرم‌افزار بوده‌اند، اما کماکان آفت مذکور در آن‌ها وجود داشته است.

پذیرفتن «بله، اما» به عنوان آفت می‌تواند منجر به ایجاد بیش و نگاهی واقع‌بینانه دراعضای تیم شود تا به روشی مهندسی برای کاهش اثرات این آفت اقدام نمایند.

پذیرش آفت «بله، اما» به این معناست که این آفت از خصوصیات رفتار آدمی است و جزء جدایی ناپذیر از توسعه سیستم‌ها است که کاهش اثرات مخرب آن نیازمند برنامه‌ریزی و کار مهندسی است. استفاده از روش‌های استخراج نیازمندی‌هایی که «اما»‌ها را زودهنگام آشکار می‌سازند می‌تواند اثرات آن را کاهش دهد. با کاربرد روش‌های مذکور، موارد «بله، اما» در مراحل ابتدایی ساخت سیستم استخراج می‌گردند. روش نمونه‌سازی^۱ یکی از روش‌های مذکور است.

-۳- آفت قلعه‌های کشف‌نشده^۲

از بسیاری جهات شناسایی نیازمندی‌ها مانند اکتشاف در قلعه‌های باستانی است. هر چه بیشتر تلاش می‌کنید بیشتر کشف می‌کنید و هر چه بیشتر کشف می‌کنید به این نتیجه می‌رسید که هنوز چیزهای زیادی برای اکتشاف باقی مانده است. هیچگاه از اتمام اکتشاف مطمئن نخواهد شد.

تیم توسعه در تلاش است تا پی‌برد که استخراج نیازمندی‌ها کی به پایان خواهد رسید و چه زمانی نیازمندی‌های کاربر به طور کامل یا به اندازه کافی استخراج شده است. این واقعیت را باید پذیرفت که استخراج نیازمندی‌ها عملیاتی است که شاید هیچگاه پایان نیابد، اما همیشه زمانی می‌رسد که می‌توان ادعا کرد «نیازمندی‌ها به اندازه لازم شناسایی شده‌اند. بقیه آن‌ها که دارای جزئیات کم‌اهمیتی هستند بعداً استخراج خواهند شد». تشخیص این زمان کار ساده‌ای نیست. تسلط بر حوزه مسئله و تجربه قبلی به این امر کمک می‌کند.

در پرورزه‌های قیمت ثابت^۳ که بر اساس درخواست برای پیشنهاد یا RFP^۴ با زمان و قیمت ثابت منعقد می‌گردد آفت قلعه‌های کشف‌نشده به مراتب چالش‌زاور است. کارفرما سعی در افزایش محوطه قلعه باستانی موضوع پرورژه و تیم توسعه سعی در کاهش آن دارد.

روش تحلیل مسئله و روش‌های استخراج نیازمندی‌ها به رفع این معضل کمک می‌کنند که در فصل‌های سوم و دهم ارائه شده‌اند.

1. Prototyping

2. The "Undiscovered Ruins" Syndrome

3. Fixed price projects

4. Request for proposal

۴- آفت کاربر- توسعه‌دهنده^۱

روش‌های استخراج نیازمندی‌ها، روش‌های جدیدی نیستند. تیم‌های توسعه در طول بیش از پنجاه سال کوشیده‌اند تا اشکالات موجود را رفع و کار خود را به شکل بهتری انجام دهنند. با این حال چرا شناسایی نادرست نیازهای ذینفعان کماکان یکی از بزرگترین مشکلات توسعه نرم‌افزار است؟

یکی از مهمترین دلایل وجود این مشکل، ناآگاهی توسعه‌دهنگان از روش‌های استخراج نیازمندی‌ها است. تفاوت فکری کاربر و توسعه‌دهنده منجر به بروز شکاف در تعامل آن‌ها می‌شود. این شکاف منجر به بروز آفت «کاربر- توسعه‌دهنده» می‌گردد. کاربران و توسعه‌دهنگان از دو دنیای جداگانه با زبان‌ها، پیش‌زمینه‌های فکری، اهداف و انگیزه‌های متفاوت درباره سیستم مذاکره می‌کنند. مشابه این تفاوت در سایر زمینه‌ها نیز وجود دارد. به عنوان مثال دکتر جان‌گری نویسنده کتاب «مردان مریخی و زنان ونسی» درباره تفاوت زنان و مردان می‌نویسد:

«فرض کنید مردها از سیاره مریخ آمدند و زن‌ها از سیاره ونس. مریخی‌ها پشت تلسکوپ‌های خود بودند که ونسی‌ها را دیدند و از آن‌ها خوشناسان آمد و با سفینه‌های خود به سوی آن‌ها رفتند. ونسی‌ها هم با آغوش باز از آن‌ها استقبال کردند. سپس با هم به زمین آمدند. مدتی همه چیز خوب بود اما کم کم هوای زمین روی آن‌ها اثر کرد و یک روز صبح وقتی بیدار شدند، هم مریخی‌ها و هم ونسی‌ها دیگر فراموش کرده بودند که از کجا آمدند و از آن روز اختلافات آن‌ها شروع شد. زن‌ها و مردها بدون توجه به آنکه قرار است با هم تفاوت‌هایی داشته باشند، رو در روی یکدیگر ایستادند. همه ما در موقعی از همسرمان دلگیر می‌شویم، زیرا این حقیقت مهم را فراموش کرده‌ایم. انتظار داریم همسر، یعنی جنس مخالف ما، آنچه را ما می‌خواهیم او نیز بخواهد و آنچه را احساس می‌کنیم، او نیز احساس کند.

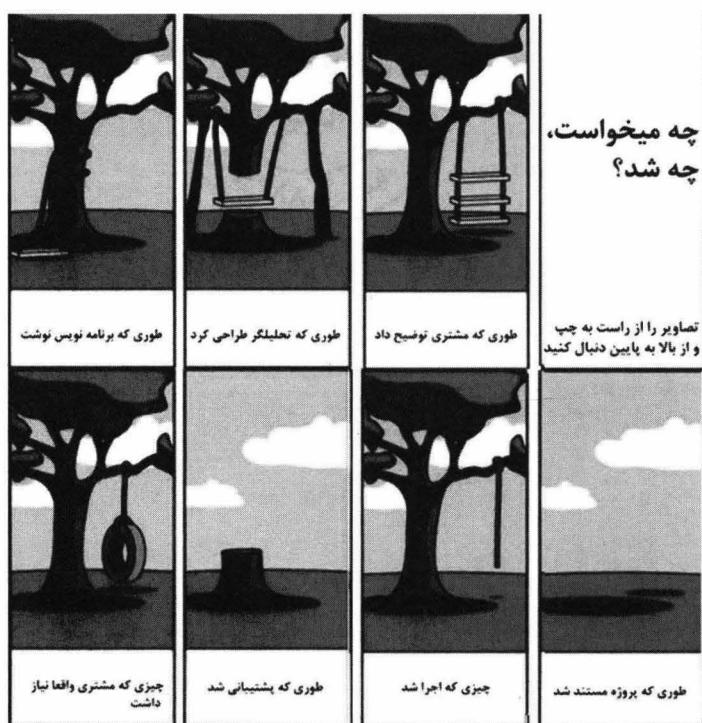
مردها به اشتباه فکر می‌کنند زن‌ها باید مانند آن‌ها بیندیشند و ارتباط برقرار کنند و زن‌ها هم به اشتباه فکر می‌کنند مردها باید مانند آن‌ها احساس کنند. اگر بتوانیم این تفاوت‌ها را بشناسیم و به آن‌ها احترام بگذاریم، تا حدود زیادی از اختلافاتمان کم می‌شود. وقتی به یاد بیاوریم که مردها از مریخ و زنها از ونس آمده‌اند، همه چیز روشن می‌شود.» (گری، ۱۳۸۶)

اگر توصیه آقای جان‌گری را پذیریم پذیرش تفاوت‌های توسعه‌دهنده و کاربر نیز تا حدود زیادی اختلاف طرفین را کاهش خواهد داد. به دنبال پذیرش موضوع لازم است به کمک روش‌های مهندسی، سعی در رفع آن گردد. شارر (Scharer, 1981) راهنمایی‌هایی برای کاهش اثرات آفت مذکور ارائه کرده است که در جدول ۱-۹ آورده شده‌اند.

جدول ۹: راه حل‌هایی برای رفع آفت کاربر - توسعه دهنده

راه حل	مسئله
کاربر را به عنوان «خبره حوزه مسئله» به رسمیت شناخته، از او بابت همکاری و ارائه تجربه و اطلاعاتی، سپاسگزار باشد. به علاوه از روش‌های استخراج نیازمندی‌ها ^۱ برای برقراری ارتباط مؤثر استفاده کنید.	کاربر دقیقاً نمی‌داند چه می‌خواهد یا اگر هم می‌داند، قادر به بیان آن نیست.
از روش‌های استخراج نیازمندی‌ها مانند ایفای نقش و نمونه‌سازی برای اطمینان از درستی خواسته‌های کاربران استفاده کنید.	کاربران تا قبل از مشاهده سیستم از درستی خواسته‌های خود اطمینان کامل دارند. اما پس از آن به اشتباه خود پی‌می‌برند.
با استفاده از روش ایفای نقش، تحلیل‌گران را برای مدتی وادر کنید تا به جای کاربر، کارهای وی را انجام دهند.	تحلیل‌گران فکر می‌کنند مشکلات و نیازهای کاربران را بهتر از آن‌ها می‌دانند.

شکل ۲-۹ تصویر معروفی است که برداشت‌های متفاوت افراد را نسبت به موضوعی واحد نشان می‌دهد.



شکل ۹: چه می‌خواستیم، چه شد

۱- این تکنیک‌ها در فصل دهم تشریح شده‌اند.

۵- نکات کلیدی

- وجود آفت‌های «بله، اما»، «قلعه‌های کشف‌نشده» و «کاربر- توسعه‌دهنده» یکی از علل دشواری استخراج نیازمندی‌ها است.
- آفت «بله، اما» از خصوصیات آدمی است و از نبود امکان تجربه کار با نرم‌افزار همانند یک ابزار فیزیکی قبل از تولید محصول نهایی ناشی می‌گردد.
- شناسایی نیازمندی‌ها مانند اکتشاف «قلعه‌های کشف‌نشده» است؛ با اکتشاف بیشتر به این نتیجه می‌رسید که بخش بیشتری را کشف نکرده‌اید.
- آفت «کاربر- توسعه‌دهنده» نمایانگر تفاوت دیدگاه کاربر و توسعه‌دهنده است که تعامل بین آن‌ها را دشوار می‌کند.

٦- مراجع

1. Leffingwell, Dean, Don Widrig. Managing Software Requirements: A Use Case Approach, Second Edition. Addison Wesley, 2003
 2. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>
 3. Scharer, Laura. "Pinpointing Requirements." 1981

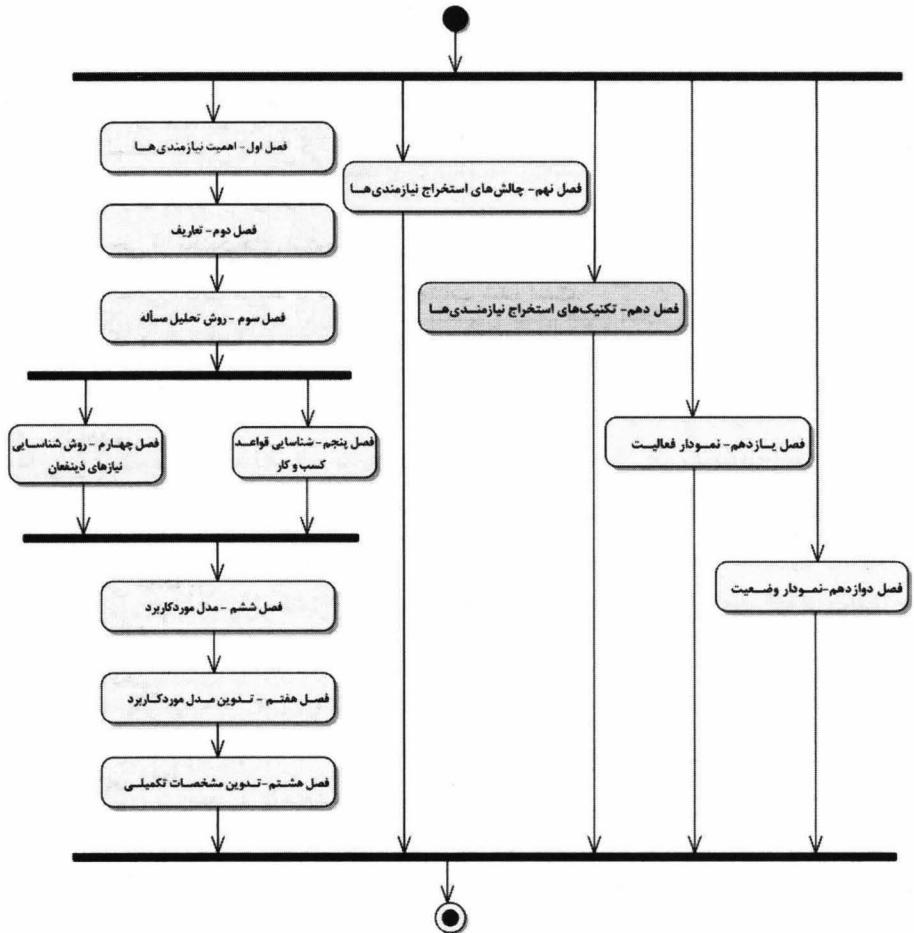
۴. گری، جان. مردان میریخی، زنان و نوسری. ترجمه مهدی قراچه داغی. آسیم، ۱۳۸۶.



فصل دهم - تکنیک های استخراج نیازمندی ها

حتی بهترین مستندات نیازمندی ها نمی توانند و نباید جایگزین گفتگوهای انسانی شوند.

کارل ای ویکرز نویسنده کتاب *More About Software Requirements*



۱- مقدمه

تکنیک‌های استخراج نیازمندی‌ها بخشی از مهندسی نیازمندی‌ها است که تیم‌های توسعه به خصوص تحلیل‌گران از آن غافل می‌شوند. اهمیت تکنیک‌ها به چگونگی استخراج نیازمندی‌ها و کیفیت آن‌ها مربوط می‌گردد. تحلیل‌گر بنابر موقعیت، یک یا ترکیبی از چند تکنیک را برای استخراج نیازمندی‌ها انتخاب می‌کند. تسلط نداشتن تحلیل‌گر به تکنیک‌ها، باعث جمع‌آوری نامطلوب نیازمندی‌ها یا بروز چالش‌هایی در اجرای پروژه‌ها می‌گردد.

این فصل، برخی از تکنیک‌های استخراج نیازمندی‌ها را معرفی می‌کند. برای پرهیز از طولانی شدن فصل، از تشریح تمامی نکات مرتبط با تکنیک‌ها اجتناب شده است.

تکنیک‌هایی که در این فصل بررسی می‌شوند، عبارتند از:

- مصاحبه
- مشاهده
- تحقیق / پرسشنامه
- تحلیل مستندات
- مهندسی معکوس^۱
- نمونه‌سازی^۲
- توفان ذهنی^۳
- گروه متمرکر^۴
- شناسایی رابط^۵
- نقالی^۶
- ایفای نقش^۷
- کارگاه نیازمندی‌ها^۸

یادآوری می‌شود که مورد کاربرد نیز یکی از تکنیک‌های استخراج نیازمندی‌ها است که در این کتاب به تفصیل آورده شده است.

1. Reverse engineering

2. Prototyping

3. Brainstorming

4. Focus group

5. Interface identification

6- Storyboarding- واژه نقالی به این دلیل برگزیده شد که یادآور نقل داستان‌های کهن پارسی توسط نقال به کمک پرده‌ای حاوی تصاویری از داستان باشد.

7. Storyboarding

8. Role playing

9. Requirements workshop

۲- مصاحبه

مصاحبه روشی ساده برای استخراج نیازها از کاربران و ذینفعان سیستم است. این تکنیک، شیوه‌ای سیستمی برای استخراج اطلاعات از فرد یا گروهی از افراد در یک جلسه رسمی یا غیررسمی است. مصاحبه به دو دسته کلی «بسته» و «باز» تقسیم می‌گردد. در مصاحبه بسته^۱ (ساخت یافته^۲)، مصاحبه‌گر پرسش‌هایی از پیش‌آمدهای در اختیار دارد و به دنبال پاسخ آن‌ها است. در مصاحبه باز^۳ (بدون ساختار)، بدون وجود پرسش از پیش طراحی شده‌ای، مصاحبه‌گر و مصاحبه‌شونده در مورد موضوع به بحث می‌پردازند.

مرز مشخصی بین این دو نوع مصاحبه وجود ندارد. عموماً مصاحبه با مجموعه پرسش‌های مشخصی شروع می‌شود و پاسخ‌های آن‌ها، زمینه بحث و پرسش‌های بعدی را فراهم می‌کند. این فرایند می‌تواند به شکل معکوس نیز اجرا گردد، یعنی ابتدا مصاحبه به شکل باز شروع شود و بعد به سوی مجموعه‌ای از پرسش‌های مشخص هدایت گردد.

انتخاب مصاحبه‌شونده یکی از مهم‌ترین گام‌ها در اجرای این تکنیک است. اولویت با کسی است که بتوان نیازهای صحیح را به کمک وی استخراج کرد. معمولاً بهترین انتخاب‌ها در سازمان کسانی هستند که اصلاً اسمی از آن‌ها در جایی برده نمی‌شود.

آشنایی با مهارت‌هایی مانند گوش دادن مؤثر^۴، زبان بدن^۵ و ارتباط مؤثر در اجرای این تکنیک بسیار اثربخش است.

مصاحبه مانند روابط اجتماعی انسان‌ها دارای پیچیدگی، دشواری و ظرافت‌های بسیاری است و موفقیت آن به مجموعه وسیعی از عوامل وابسته است. سطح تسلط مصاحبه‌گر به حوزه مسأله، انتخاب درست مصاحبه‌شونده و آمادگی وی برای پاسخ‌گویی، تجربه مصاحبه‌گر در هدایت و پیش‌برد مصاحبه، مهارت مصاحبه‌گر در مستندسازی مباحث مصاحبه و چگونگی تعامل طرفین از ابتدایی‌ترین عوامل هستند.

تنظیم روش پاسخ‌گویی در مصاحبه گروهی، تحلیل انبوه اطلاعات حاصل از مصاحبه و بروز تناقض بین اظهارات ذینفعان از دشواری‌های این تکنیک است. با این وجود، این تکنیک رایج‌ترین تکنیک استخراج نیازمندی‌ها در توسعه سیستم‌های نرم‌افزاری است.

1. Closed Interview
2. Structured
3. Open Interview
4. Active listening
5. Body language

۳- مشاهده

مشاهده ابزاری برای استخراج نیازمندی‌ها بر اساس دیدن کار افراد در محیط واقعی است.

مشاهده می‌تواند مستقیم با حضور در محیط یا غیرمستقیم از طریق رسانه‌ها از جمله فیلم انجام شود.

مشاهده مستقیم امکان تمرکز بر بخشی از محیط را فراهم می‌سازد، در حالی که در مشاهده غیرمستقیم، بر بخش‌هایی که در مشاهده مستقیم به آن‌ها توجه نشده است، تمرکز می‌شود. به علاوه در این روش، امکان تکرار (فیلم یا رسانه) نیز فراهم است.

رویکردهای «فعال» و «غیرفعال» دو رویکرد اساسی در اجرای تکنیک مشاهده هستند. در رویکرد

غیرفعال یا نامرئی، مشاهده گر تنها نظاره‌گر است و گفتگویی با مجری ندارد. در رویکرد فعال یا مرئی، مشاهده گر با مجری به گفتگو و پرسش و پاسخ می‌پردازد.

این شیوه دانشی واقع‌بینانه و عملی از چگونگی انجام کار ایجاد می‌کند و امکان تعامل مؤثرتر تیم

توسعه و کاربران نهایی را فراهم می‌سازد که می‌تواند در ادامه پروژه تأثیری مطلوب داشته باشد.

اجرای این شیوه تنها برای فرایندهای موجود امکان‌پذیر است. زمان بر بودن، احتمال ایجاد مزاحمت،

بی‌نظمی و سربارهای آماده‌سازی در محیط و احتمال بسیار ضعیف مشاهده شرایط بحرانی و استثناء از معایب این روش است.

روش مشاهده، در استخراج نیازمندی‌ها روشی بسیار کارآمد است. در کنار تمامی مزیت‌های این

روش، تغییر تصور ذهنی تحلیل گر به دانش مبنی بر شرایط محیط واقعی، مهم‌ترین دستاورده آن است.

۴- تحقیق / پرسش نامه

تحقیق ابزاری برای استخراج اطلاعات از تعداد زیادی از افراد و در زمانی کوتاه به کمک یک پرسشنامه است.

پرسش‌های تحقیق می‌تواند «بسته» یا «باز» باشد. در نوع بسته، پاسخ‌دهنده بایستی از میان پاسخ‌های ارائه شده، یک یا چند مورد را انتخاب نماید. این نوع پرسش وقتی مفید است که موضوعات، مشخص ولی دامنه پاسخ‌های کاربران نامشخص است. در نوع باز، پاسخ‌دهنده در پاسخ به پرسش‌ها آزاد است. این نوع پرسش زمانی مفید است که دامنه پاسخ‌ها نامشخص باشد.

پرسش‌های باز، چشم‌انداز و دیدگاه‌های پاسخ‌دهنده‌گان را بهتر منعکس می‌کنند.

پرسشنامه‌ها را می‌توان بر اساس روش اجرا نیز دسته‌بندی کرد. پرسشنامه همراه با مصاحبه،

پرسشنامه‌ای است که به صورت حضوری از افراد پرسیده می‌شود و پاسخ‌ها در پرسشنامه وارد می‌گردد. پرسشنامه خود اظهار، پرسشنامه‌ای است که در اختیار فرد یا گروه قرار می‌گیرد و پاسخ‌گو به تنهایی یا گروهی اقدام به تکمیل پاسخ‌نامه می‌کند. پرسشنامه‌های پستی از طریق پست یا پست

الکترونیکی ارسال می‌شود و پس از تکمیل، پاسخ‌نامه بازگردانده می‌شود. در پرسش‌نامه الکترونیکی، پاسخ‌دهندگان از طریق اینترنت یا شبکه‌های محلی اقدام به پاسخ‌گویی می‌کنند.

استفاده از تکنیک مصاحبه قبل از تحقیق، ایده‌های خوبی برای پرسش‌ها فراهم می‌سازد و استفاده از آن پس از تحقیق می‌تواند منجر به طرح پرسش‌های دقیق‌تر گردد.

تعداد پرسش‌های تحقیق نباید زیاد باشد تا پاسخ‌گویی به آن‌ها در زمانی کوتاه میسر باشد. در طراحی پرسش‌ها باید از طرح پرسش‌های با معرفتی، پیچیده و پرسش‌هایی که پاسخ‌دهنده را ناراحت می‌کند، اجتناب شود. تلاش برای استخراج اطلاعاتی که به صورت قانونی ممنوع است، پاسخ‌دهنده را در حالت تدافعی قرار خواهد داد.

بعد از جمع‌آوری پاسخ‌نامه‌ها، پاسخ‌ها ارزیابی، تلفیق و نتایج، تحلیل و خلاصه‌سازی می‌گردد. استفاده از روش تحقیق در استخراج نیازمندی‌ها، عملاً زمان زیادی از پاسخ‌دهندگان نمی‌گیرد. این شیوه، زمانی که ذینفعان در دسترس نباشند یا دسترسی به آن‌ها به سختی امکان‌پذیر باشد (به دلیل عواملی چون توزیع جغرافیایی، هزینه یا موضع سازمانی)، کارآمد است. طراحی پرسش‌نامه مناسب و اجرای تحقیق، کاری بسیار پیچیده و طاقت‌فرسا است.

در مجموع، این شیوه برای جمع‌آوری نیازمندی‌های سیستم روش مناسبی نیست.

۵- تحلیل مستندات

تحلیل مستندات روشی برای استخراج نیازمندی‌ها و اطلاعات با مطالعه و بررسی مستندات است. روال استخراج شامل مطالعه و تحلیل مستنداتی نظیر طرح‌های کسب‌وکار، مطالعات بازار، قراردادها، درخواست‌های پیشنهاد^۱، فرایندهای کسب‌وکار، راهنمای کاربری سیستم‌های موجود، راهنمای آموزش، مقایسه محصولات و گزارش مشکلات است.

اطمینان از صحت و بهروز بودن مستندات در این روش بسیار حائز اهمیت است. در این روش، از داده‌ها و اطلاعات موجود در مستندات می‌توان به عنوان ابزاری برای بررسی درستی، تأیید یا رد نیازمندی‌های استخراج شده از سایر تکنیک‌ها نظیر مصاحبه، مشاهده و پرسش‌نامه استفاده کرد.

محدود بودن به محتوای مستندات موجود، بی‌اعتباری و بهروز نبودن مستندات، زمان‌بر و گاه کسل‌کنندگی از ایرادات آن به شمار می‌آید.

۶- مهندسی معکوس^۱

مهندسی مستقیم، فرایند سنتی حرکت از سطوح بالا به پیاده‌سازی سیستم یا ساخت یک دستگاه است. در مقابل، مهندسی معکوس فرایند اکتشاف فناوری‌های به کار رفته در یک دستگاه یا سیستم از طریق تحلیل ساختار، عملکرد و وظیفه آن است.

مهندسي معکوس به دو گونه دربسته^۲ و درباز^۳ تقسیم می‌گردد. مهندسی معکوس در جعبه دربسته بدون توجه به ساختار درونی و در درباز با بررسی ساختار درونی سیستم یا محصول انجام می‌گردد. این روش زمانی مناسب است که سیستم موجود دارای مستندات ناچیز و نامعتبری باشد. اغلب از این روش برای دسترسی به داشت نهفته در محصول رقبا و تحلیل عملکرد سیستم‌های قدیمی بدون مستندات استفاده می‌گردد.

مهندسي معکوس اغلب گران، زمانبر و نیازمند مهارت‌های تخصصی ویژه‌ای است. از این‌رو تحلیل هزینه‌فایده برای انجام آن الزاماً است. به علاوه، این روش اغلب ناقض قوانین حق تکثیر یا نشر(کپی رایت) است.

۷- نمونه‌سازی^۴

نمونه اولیه، نسخه ابتدایی از سیستمی است که در مراحل آغازین توسعه قرار دارد. یکی از کاربردهای نمونه‌سازی در سیستم‌های نرم‌افزاری، استخراج و اعتبارسنجی نیازمندی‌ها است. در این روش، نمونه‌ای از رابط کاربر تهیه می‌گردد و با سایر نیازمندی‌ها نظیر موردنی کاربرد و قواعد کسب و کار مطابقت داده می‌شود.

نمونه اولیه را از جنبه‌های مختلفی می‌توان دسته‌بندی کرد. بر اساس دامنه عملکرد، نمونه اولیه به دو دسته «نمونه اولیه افقی» و «نمونه اولیه عمودی» تقسیم می‌گردد. در نمونه اولیه افقی، سطحی کم عمق و احتمالاً وسیع از سیستم بدون پیاده‌سازی قاعده یا کنترلی تهیه می‌شود. در نمونه اولیه عمودی، سطحی عمیق و اغلب محدود با رعایت قواعد و کنترل‌ها پیاده‌سازی می‌گردد.

بر اساس چرخه عمر نمونه اولیه، می‌توان آن را به دو دسته دورریختنی و تکاملی دسته‌بندی نمود. در «نمونه اولیه دورریختنی»، رابط کاربر با استفاده از ابزاری ساده، گاه مداد و کاغذ تهیه می‌گردد. چنین نمونه‌ای بعد از این مرحله دور ریخته می‌شود. در «نمونه اولیه تکاملی» بخشی از سیستم نمونه‌سازی می‌شود و پس از دریافت نظرات کاربران و اعمال آن‌ها، بخش دیگری نیز به نمونه افزوده می‌شود تا در آخر به محصول نهایی تبدیل گردد.

بیشترین نمونه‌های ساخته شده در مرحله نیازمندی‌ها، افقی، دورریختنی و برای رابط کاربر سیستم

1. Reverse engineering

2. Black Box

3. White Box

4. Prototyping

هستند. این تکنیک یکی از بهترین تکنیک‌های استخراج نیازمندی‌های کاربران است. نمونه‌سازی با مشارکت کاربران به آن‌ها امکان می‌دهد تا نمونه سیستم را مشاهده و نظرات خود را ارائه کنند. مشاهده نمونه توسط کاربر از اغلب سوء‌برداشت‌ها جلوگیری خواهد کرد و به بهبود ارتباط بین کاربر و توسعه‌دهنده و رفع آفت‌های «کاربر-توسعه‌گر» و «بله؛ اما» کمک می‌کند.

باید توجه داشت که معمولاً تشکیل تیمی برای نمونه‌سازی هزینه‌بر است. گاه بسنده کردن به نمونه اولیه باعث غفلت توسعه‌دهنده‌گان از تحلیل مناسب سیستم می‌گردد. از سوی دیگر، تلاش تیم توسعه‌دهنده برای ساخت نمونه اولیه، باعث علاقه‌مندی و شیفتگی آنان نسبت به نمونه می‌گردد تا جایی که آن را به عنوان محصول نهایی پذیرفته، تلاشی برای تغییر آن نمی‌کنند.

۸- توفان ذهنی^۱

توفان ذهنی روشی مؤثر به منظور شناسایی راه حل‌های خلاقانه برای یک موضوع است. این روش توسط الکس آزبورن در سال ۱۹۴۸ معرفی گردید. آزبورن می‌گوید: «پیشنهاد خلق شده در ذهن فرد حاضر در یک گروه، دو برابر پیشنهاد خلق شده وی در تنهایی است». در آن زمان بنیاد فرهنگی آزبورن این روش را در چندین شرکت تحقیقاتی، بازرگانی، علمی و فنی برای حل مشکلات و مسائل مدیریت به کار گرفت. موفقیت این روش در کمک به حل مسائل آن چنان بود که ظرف مدت کوتاهی به عنوان روشی کارآمد شناخته شد.

توفان ذهنی شامل دو مرحله است، «تولید ایده‌ها» که طی آن ایده‌ها بیان و جمع‌آوری می‌شوند و «ارزیابی یا کاهش ایده‌ها» که در آن، ایده‌های جمع‌آوری شده ارزیابی می‌شوند.

در مرحله تولید ایده‌ها، محدودیتی برای تعداد ایده‌ها وجود ندارد و باید تمام ایده‌ها ثبت گردند. با اتمام مرحله تولید، ایده‌ها ارزیابی شده، با حذف موردهای تکراری، ترکیب و رتبه‌بندی، فهرست نهایی آمده می‌گردد. برای رتبه‌بندی، روش‌های متعددی به کار می‌رود که ساده‌ترین آن‌ها رأی‌گیری است. اغلب ایده‌های نهایی، ایده‌های یک شرکت‌کننده نیست، بلکه ترکیبی از ایده‌های چندین شرکت‌کننده است که تأییدی است بر گفته بلانچارد که «هیچ یک از ما باهوش‌تر از همه ما نیست».

به طور خلاصه می‌توان گفت که توفان ذهنی مبتنی بر قواعد زیر است:

- پرهیز از نقد ایده یا بحث درباره آن
- اهمیت کمیت و تعداد ایده‌ها
- استقبال از ایده‌های غیرمعمول
- ترکیب و بهبود ایده‌ها برای خلق ایده جدید

محیط غیرقضاؤی این روش، امکان تفکر خارج از چارچوب قواعد رایج را برای افراد فراهم

می‌آورد. این کار موجب ظهور ایده‌ها و راه حل‌های خلاقانه و بدیع می‌گردد. این روش بسیار وابسته به خلاقیت شرکت‌کنندگان است و حضور افرادی از سطوح بالای ساختار سازمانی می‌تواند در رد، قبول، نقد و حتی ارائه ایده‌ها اثرگذار باشد. در چنین شرایطی توصیه می‌گردد از روش گروه صوری^۱ به جای توفان ذهنی استفاده گردد.

۹- گروه متمرکز^۲

گروه متمرکز شیوه‌ای برای تبادل افکار و مذاکره مطابق با برنامه‌ای مشخص است که در آن، نظرات و دیدگاه شرکت‌کنندگان سنجیده می‌شود. این گروه باید متشکل از افرادی متجانس و همگن، بدون آشنایی قابلی، با حداقل تأثیرپذیری از دیگران و نمونه‌ای از گروه یا جامعه بزرگتر باشند.

شرکت‌کنندگان در این گونه جلسات آزادانه دغدغه‌ها، اولویت‌ها و نیازهای خود را در خصوص ویژگی‌های سیستم یا نمونه‌ای از آن مطرح می‌کنند. این روش در یافتن پاسخ پرسش‌های شبیه به موارد زیر کارست:

○ کاربران درباره سیستم چگونه فکر می‌کنند؟

○ چه بخش‌هایی از سیستم برای کاربران مهم است؟

○ خواسته‌ها و انتظارات کاربران از سیستم چیست؟

این روش، روشی کیفی است که خروجی آن به جای یک گزارش آماری و عددی، گزارشی تحلیلی- تشریحی از دیدگاه‌های افراد است.

جلسات گروه متمرکز قابلیت استخراج اطلاعات از گروهی از افراد را در یک زمان کوتاه فراهم می‌آورد که در قیاس با مصاحبه، زمان و هزینه کمتری را در بر دارد. این جلسات، محیطی را برای بازنگری افکار و آشنایی با نگرش و خواسته‌های دیگران برای شرکت‌کنندگان فراهم می‌کند.

در این جلسات ممکن است شرکت‌کنندگان دغدغه اعتماد به سایرین را داشته باشند و اطلاعات شخصی خود را بیان نکنند. از سوی دیگر، ممکن است اطلاعات ارائه شده افراد با آنچه که به آن عمل می‌کنند، منطبق نباشد. به علاوه در گروه‌های همگن و یکدست، ممکن است مجموعه‌ای کامل و متنوع از نیازها مطرح نگردد.

روش گروه متمرکز شبیه به روش توفان ذهنی است، اما این دو دارای تفاوت‌های بنیادی هستند. افراد شرکت‌کننده، میزان ساخت‌یافتنگی، برنامه‌ریزی جلسات، چارچوب و اهداف آن‌ها با هم تفاوت دارند.

1. Nominal group technique
2. Focus group

۱۰- شناسایی رابط^۱

سیستم‌ها معمولاً^۱ به رابط‌های زیر نیاز دارند:

- رابط کاربر

کاربران از رابط کاربر برای کار با سیستم استفاده می‌کنند. فرم‌ها و گزارش‌ها از این دسته هستند.

- رابط سیستمی

رابط سیستمی، رابطی است که هر سیستم برای سرویس‌دهی به سایرین ارائه می‌کند.

- رابط سخت‌افزاری

رابطهایی که برای استفاده از تجهیزات سخت‌افزاری در اختیار سیستم قرار می‌گیرد، رابط سخت‌افزاری نامیده می‌شود.

شناسایی رابطهای لازم برای پشتیبانی از عملکرد سیستم، علاوه بر کمک به تعیین مرز آن، موجب استخراج نیازمندی‌های فراوانی نیز می‌گردد. پس از شناسایی رابطهای جزئیات هر یک استخراج و مستند می‌شود.

این تکنیک دیدی کلان و زودهنگام از ارتباط اجزای سیستم فراهم می‌کند. این روش به تنها برای درکی از جریان داده‌ها و کنترل ارائه نمی‌دهد و تنها به شناسایی پروتکل ارتباطی، ورودی و خروجی بستنده می‌کند.

۱۱- نقالی^۲

نقالی (دانستان‌گویی) گام‌های یک سناپیو را به شکل مجموعه‌ای از تصاویر، نمودار یا نماهای پس‌درپی نمایش می‌دهد. این تکنیک در تیمهای تولید کارتون و پویانمایی^۳ کاربرد فراوانی دارد. ابتدا تصاویر اولیه‌ای بر تخته سفید یا سیاه چسبانده می‌شوند و قصه بر اساس آن نقل و در صورت نیاز اصلاح می‌گردد. بعد از توافق، تولید کارتون یا پویانمایی شروع می‌شود.

نقالی در توسعه نرم‌افزار، ابزاری برای نمایش کاربران سیستم، چگونگی رفتار سیستم و جایگاه سیستم در سازمان کارفرما است. نقالی می‌تواند شامل سناپیوهایی باشد که قرار نیست در سیستم پوشش داده شوند. ابزار نقالی می‌تواند نرم‌افزار پاورپوینت، فلاش و حتی مداد و کاغذ باشد. صفحه نقالی (محتوای فایل فلاش، پاورپوینت و کاغذ) باید ساده و قابل تغییر باشد و اگر نتوان آن را سریع تغییر داد، نمی‌توان نتایج مناسبی از آن به دست آورد.

نقالی تمامی مزیت‌ها و کاستی‌های تکنیک‌های تصویری مانند نمونه‌سازی را دارد. نکته بارز در نقالی

1. Interface identification
2. Storyboarding
3. Animation

این است که بر خلاف نمونه‌سازی، می‌توان سناریوهایی مرکب از موضوعات غیرنرم‌افزاری مانند فرایند و محیط کسب‌وکار را با تصاویری از نرم‌افزار ترکیب کرد تا بینده تصویری جامع از نرم‌افزار به دست آورد.

۱۲- ایفای نقش^۱

ایفای نقش روشنی است گروهی که در آن، سناریویی با ایفای نقش توسط اعضای گروه اجرا می‌شود. در این روش، هر عضو گروه بازی یک یا چند نقش را به عهده می‌گیرد. نقش‌ها می‌توانند کاربران، سیستم، سایر سیستم‌ها و گاه موجودیت‌های مهم باشند.

در مسیر اجرای سناریو، بحث‌هایی از قبیل «چه کسی مسئول چه کاری است» انجام و درباره آن تصمیم‌گیری می‌شود. موارد کاربرد سیستم، مبنای مناسبی برای تعریف سناریوها هستند.

این روش می‌تواند برای تعیین دامنه، شناسایی و اولویت‌بندی نیازمندی‌ها به کار رود.

این تکنیک با شبیه‌سازی، وضعیت نهایی را برای کاربران و وضعیت فعلی را برای تیم تحلیل ملموس می‌کند که خود باعث استخراج نیازها می‌گردد. این شیوه معمولاً کم‌هزینه است. موقوفیت این تکنیک در گرو خبرگی اعضا و دبیر جلسات آن است. معمولاً از تکنیک کارت‌های سی‌آرسی^۲ نیز برای ثبت مسئولیت‌های هر نقش استفاده می‌شود.

۱۳- کارگاه نیازمندی‌ها^۳

به جرأت می‌توان گفت که کارگاه نیازمندی‌ها قدرتمندترین تکنیک برای استخراج نیازمندی‌های است. مهم‌ترین ویژگی این تکنیک، گردهم‌آوری تمامی ذینفعان در یک زمان کوتاه است. کارگاه می‌تواند برای تعیین دامنه، شناسایی، اولویت‌بندی، بازنگری و تأیید نیازمندی‌ها از جمله ویژگی‌ها و موارد کاربرد به کار گرفته شود.

در مراحل ابتدایی پروژه، کارگاه با هدف استخراج نیازهای ذینفعان برگزار می‌گردد. در ادامه، کارگاه با هدف بازنگری و تأیید موارد کاربرد شناسایی شده برگزار می‌شود و در پی آن، هدف کارگاه به مرور و تأیید نهایی نیازمندی‌ها تغییر می‌کند.

توجه داشته باشید که کارگاه نیازمندی‌ها با جلسه‌های عمومی تفاوت‌های فاحشی دارد. لذا هر جلسه‌ای که با حضور ذینفعان در یک زمان مشخص برگزار می‌گردد، لزوماً کارگاه نیازمندی‌ها نیست. کمتر بودن زمان و هزینه کارگاه نسبت به تکنیک‌هایی چون مصاحبه، رفع معضل تعارض در گفته‌های ذینفعان به خصوص در مصاحبه‌های انفرادی و توافق داخلی ذینفعان برای اولویت‌دهی و تأیید نیازمندی‌ها از مهم‌ترین دستاوردهای این تکنیک است.

1. Role playing

2. CRC (Class-Responsibility-Collaboration)

3. Requirements workshop

دشواری گردهم‌آوری ذینفعان در یک زمان مشخص، وابستگی موقتی آن به خبرگی دبیر کارگاه و دانش شرکت‌کنندگان، انتخاب ذینفعان مناسب با در نظر گرفتن تبعات فنی و سازمانی آن و دشواری جلب مشارکت آن‌ها از معایب این روش است.

یکی از ویژگی‌های منحصر به فرد کارگاه نیازمندی‌ها، امکان استفاده از سایر تکنیک‌ها مانند توفان ذهنی، نقالی، ایفای نقش و موردکاربرد در حین اجرای آن است.

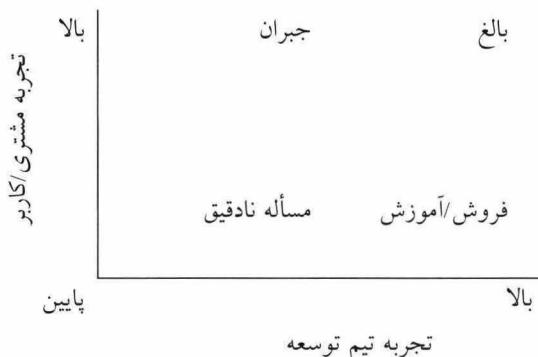
۱۴- انتخاب تکنیک مناسب

انتخاب تکنیک مناسب برای استخراج نیازمندی‌ها به عوامل متعددی بستگی دارد که از آن جمله می‌توان به موارد زیر اشاره کرد:

- محل استقرار و چگونگی دسترسی به ذینفعان
- سطح دانش تیم توسعه از حوزه مسأله
- میزان تسلط مشتریان و کاربران بر حوزه مسأله
- میزان آگاهی مشتریان و کاربران از فرایند توسعه نرم‌افزار
- ابزارهای مورد استفاده در تیم توسعه

اگر دسترسی به ذینفعان دشوار باشد (مثلاً در شهرهای دور از هم سکونت داشته باشند) یا امکان دسترسی به آنان مقدور نباشد (در بعضی از پروژه‌های خاص)، استفاده از تکنیک‌هایی که نیاز به حضور آنان دارد، دشوار و گاه غیرممکن خواهد بود.

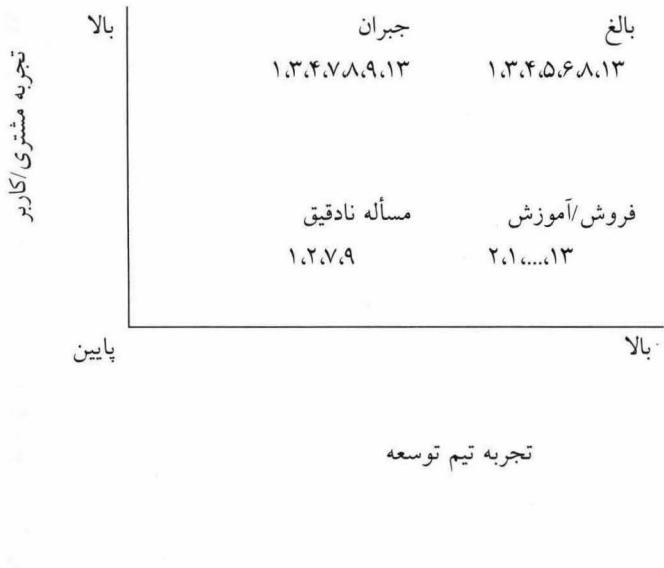
اگر ذینفعان در دسترس باشند، آن‌گاه سطح دانش ذینفعان و تیم از حوزه مسأله و فرایند توسعه نرم‌افزار، معیار اصلی برای انتخاب تکنیک خواهد بود. به عنوان مثال، اگر کاربران با موردکاربرد آشنا نباشند، اجرای کارگاه نیازمندی‌ها با هدف بازنگری موارد کاربرد، امری عبث و بیهوده خواهد بود. روش قانع‌کننده‌ای برای انتخاب تکنیک وجود ندارد، اما روش پیشنهادی زیر جالب توجه است.



شکل ۱-۱۰: ناحیه‌بندی بر اساس شرایط کاربران/مشتریان و تیم توسعه

شکل قبل بر اساس (Davis, 2000) ترسیم شده است و چارچوبی برای تعیین تکنیک مناسب ارائه می‌کند. در این تصویر، میزان مهارت‌های تیم توسعه و مشتریان/کاربران در دو محور مختصات مشخص شده است. سطح بین دو محور به چهار ناحیه تقسیم شده است: «مسئله نادقیق^۱»، «فروش/آموزش^۲»، «جبران^۳» و «بالغ^۴». به عنوان مثال ناحیه «بالغ» به شرایطی اشاره دارد که هر دو تیم توسعه و کاربران/مشتریان دارای مهارت‌های لازم و کافی برای استخراج نیازمندی‌ها هستند یا در ناحیه «نادقیق^۵» هیچ یک از طرفین دارای مهارت‌های لازم نیست.

در شکل ۲-۱۰ تکنیک‌های مناسب برای هر ناحیه پیشنهاد شده است.



شکل ۲-۱۰: ناحیه‌بندی بر اساس شرایط کاربران/مشتریان و تیم توسعه

توجه داشته باشید که با شروع فرایند و به کارگیری تکنیک‌ها در یک ناحیه، ممکن است شرایط تیم و کاربران تغییر کند و مختصات به ناحیه جدیدی در نمودار انتقال یابد. در ناحیه جدید انتخاب تکنیک مجددًا باید انجام شود.

1. Fuzzy problem
2. Selling/Teaching
3. Catch up
4. Mature
5. Fuzzy

۱۵- نکات کلیدی

- مصاحبه، مشاهده، تحقیق/پرسش‌نامه، تحلیل مستندات، توفان ذهنی، مهندسی معکوس، نمونه‌سازی، گروه متمرکز، شناسایی رابط، نقالی، ایفای نقش و کارگاه نیازمندی‌ها از مهم‌ترین تکنیک‌های استخراج نیازمندی‌ها هستند.
- یادگیری تکنیک‌ها، روش اجرا، مزایا و معایب تکنیک‌های استخراج نیازمندی‌ها برای تیم توسعه به خصوص تحلیل‌گران الزامی است.
- فرمول ساده‌ای برای انتخاب تکنیک مناسب وجود ندارد. اما بر اساس مهارت‌های تیم توسعه و کاربران/مشتریان می‌توان این تکنیک‌ها را انتخاب کرد.

۱۶- مراجع

1. Hickey, A. Davis, A. "Elicitation Technique Selection: How Do the Experts Do It." IEEE Computer Society Press, 2000
2. OpenUp. The Eclipse Foundation. <http://www.eclipse.org/epf>
3. Rational Unified Process. IBM Rational Software. 2007. <http://www-01.ibm.com/software/awdtools/rup>



بخش ششم: نمودارها

در این بخش، دو نمودار از نمودارهای زبان UML- نمودار فعالیت و نمودار وضعیت- که در تحلیل نیازمندی‌ها کاربرد بیشتری دارند، معرفی و تشریح می‌گردند.

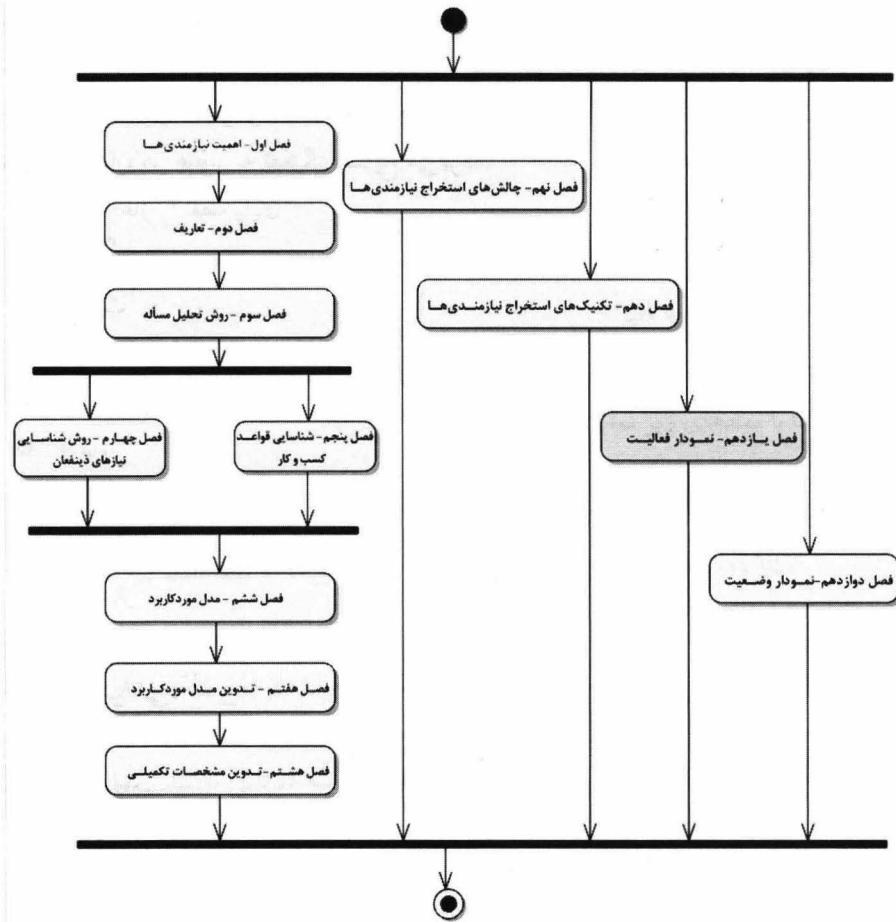
فصل یازدهم- نمودار فعالیت

فصل دوازدهم- نمودار وضعیت

فصل یازدهم - نمودار فعالیت

تنها دو نوع زبان برنامه‌نویسی وجود دارد: زبانی که همه از آن شکایت می‌کنند و زبانی که هیچ کس از آن استفاده نمی‌کند.

بیارن استروستروب، خالق C++



۱- مقدمه

نمودار فعالیت یکی از پرکاربردترین ابزارها در تحلیل سیستم است. این نمودار برای مدل‌سازی یک یا چند مورد کاربرد، عملیات پیچیده، فرایندها و قواعد کسب‌وکار به کار می‌رود. در این فصل به تشریح نمودار فعالیت پرداخته می‌شود.

مجموعه عناصری که می‌توانند در نمودار فعالیت به کار گرفته شوند، در زیر فهرست شده است که در ادامه فصل، هر عنصر به تفکیک تشریح می‌گردد.

- نقطه آغاز^۱ / نقطه پایان^۲
- عمل^۳
- فعالیت^۴
- گردش کنترل^۵
- تصمیم^۶
- ادغام^۷
- انشعاب^۸
- الحق^۹
- جداساز^{۱۰}
- شیء^{۱۱}
- گردش شیء^{۱۲}
- گیره^{۱۳}
- انباره داده^{۱۴}
- میانگیر مرکزی^{۱۵}
- منطقه توسعه^{۱۶}
- پایان گردش^{۱۷}

-
1. Starting point
 2. Ending point
 3. Action
 4. Activity
 5. Control flow
 6. Decision
 7. Merge
 8. Fork
 9. Join
 10. Partition
 11. Object
 12. Object flow
 13. Pin
 14. Data store
 15. Central buffer
 16. Expansion region
 17. Flow final

- رخداد زمانی^۱
- سیگنال^۲
- نشانه^۳
- بست^۴

برای هر عنصر، تعریف، کاربرد، نماد تصویری در UML و مثال‌هایی از آن ارائه می‌شود.

۲- عناصر نمودار فعالیت

در این بخش، عناصر و اجزای نمودار فعالیت تشریح می‌گردد.

۱-۱- نقطه آغاز^۵/ نقطه پایان^۶

«نقطه آغاز» نشان‌دهنده شروع فعالیت و «نقطه پایان» نشان‌دهنده نقطه اتمام فعالیت است. این دو عنصر در UML با نمادهای زیر نمایش داده می‌شوند.



شکل ۱-۱-۲: نقطه پایان

شکل ۱-۱: نقطه آغاز

هر نمودار فعالیت تنها یک «نقطه آغاز» دارد که معمولاً در بالای نمودار و در سمت چپ یا وسط آن قرار می‌گیرد. از آنجا که ممکن است چندین مسیر متفاوت برای خروج از نمودار وجود داشته باشد، امکان وجود بیش از یک «نقطه پایان» در نمودار فعالیت وجود دارد. در عین حال می‌توان تمام مسیرها را به یک «نقطه پایان» متصل نمود، اما برخی از اوقات به دلایلی مانند زیبایی و خوانایی نمودار، این کار انجام نمی‌شود.

۲-۲- عمل^۷

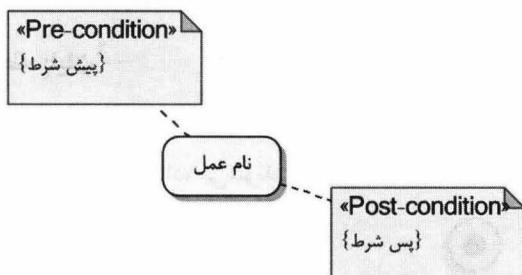
«عمل» گامی بنیادین در مدل‌سازی یک رویه است که نمایان‌گر کاری است که قابل تفکیک به کارهای کوچکتر نیست. به عبارت دیگر، گامی تجزیه‌ناپذیر در رویه مدل‌شده را نمایش می‌دهد. «عمل» در UML با نماد زیر نمایش داده می‌شود.

1. Time event
2. Signal
3. Token
4. Connector
5. Starting point
6. Ending point
7. Action

نام عمل

شکل ۱۱-۳: عمل

«عمل» می‌تواند چندین جریان ورودی و خروجی از نوع «گردش کترل» و «گردش شیء» و مجموعه‌ای از شرط‌ها به عنوان پیش‌شرط یا پس‌شرط داشته باشد. «عمل» تنها پس از تحقق تمامی ورودی‌ها و پیش‌شرط‌ها اجرا می‌گردد و پس از اجرا، شرایط توصیف شده در پس‌شرط محقق می‌گردند.



شکل ۱۱-۴: پیش‌شرط و پس‌شرط

۳-۲- فعالیت^۱

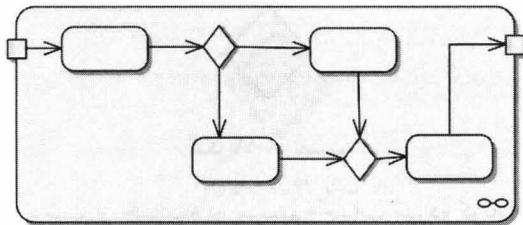
عنصر «فعالیت» نشان‌دهنده گامی تجزیه‌پذیر در رویه است. «فعالیت» نشان‌دهنده مجموعه‌ای از «عمل‌ها» است که کار مشخصی را انجام می‌دهند و در قالب «فعالیت»، یک گام از یک رویه را تشکیل داده‌اند. در UML «فعالیت» به شکل زیر نمایش داده می‌شود.

نام فعالیت

شکل ۱۱-۵: فعالیت

نماد این عنصر در UML بسیار به نماد «عمل» شبیه است. این نماد نشان‌دهنده «فعالیتی» است که «فعالیت ساده»^۲ نامیده می‌شوند. علاوه بر این نوع، «فعالیت» دیگری با نام «فعالیت پیچیده»^۳ نیز وجود دارند که به شکل زیر نمایش داده می‌شود.

1. Activity
2. simple activity
3. complex activity



شکل ۱۱-۶: فعالیت پیچیده

نماد شکل قبل در UML وجود ندارد و تصویر با ترکیب عناصر نمودار فعالیت و جای دادن آنها در «فعالیت» برای نمایش رویه، ساخته شده است تا نشان دهد که در «فعالیت پیچیده»، عناصر نمودار در خود «فعالیت» جای گرفته‌اند.

«فعالیت» و «عمل» غالباً به اشتباه به جای هم به کار می‌روند. «عمل» مانند دستورات زبان برنامه‌نویسی و «فعالیت» مانند متدها یا توابع هستند. یک دستور زبان برنامه‌نویسی، عنصری تجزیه‌نپذیر در اجرای برنامه است و متد یا تابع، مجموعه‌ای از دستورات و عناصری مرکب هستند. برای نمونه در «سیستم حساب سپرده کوتاه مدت» تعریف مشتری شامل مجموعه‌ای از گام‌ها و کارهای است. اما این کار یک گام در رویه «افتتاح حساب» است. از این رو، تعریف مشتری یک «فعالیت» در نمودار افتتاح حساب خواهد بود. این «فعالیت» شامل «عمل‌های» ورود مشخصات مشتری، ذخیره مشخصات مشتری و صدور شماره مشتری است.

۴-۴- گردش کنترل^۱

«گردش کنترل»، ارتباط‌دهنده دو عنصر موجود در نمودار فعالیت است که پس از اتمام اجرای یک عنصر، کنترل را به عنصر دیگر منتقل می‌کند. در UML این عنصر به شکل پیکانی مطابق شکل زیر نمایش داده می‌شود.



شکل ۱۱-۷: گردش کنترل

۵-۲- تصمیم^۲

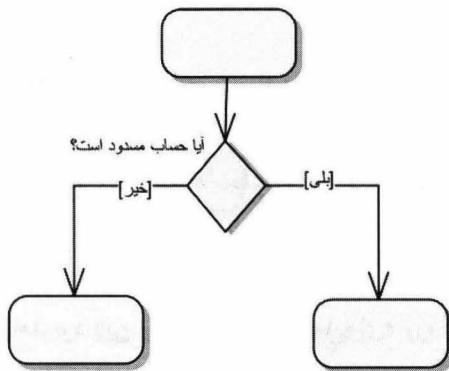
عنصر «تصمیم» نقاطی را در نمودار فعالیت نشان می‌دهد که در آن نقاط بر اساس یک شرط، از بین دو یا چند مسیر برای ادامه رویه، تنها یک مسیر باید انتخاب گردد. این عنصر به شکل زیر نمایش داده می‌شود.

1. Control flow
2. Decision



شکل ۱۱-۸: تصمیم

برای مثال در رویه «برداشت از حساب» در سیستم «حساب سپرده کوتاه مدت» باید قبل از عمل برداشت، شرط «آیا حساب مسدود است؟» بررسی شود. اگر حساب مسدود باشد، روال کار با شرایطی که حساب مسدود نباشد، متفاوت است. برای نشان دادن این شرایط و تصمیم‌گیری برای ادامه رویه، از عنصر «تصمیم» استفاده می‌شود.



شکل ۱۱-۹: نمونه تصمیم

بر روی «گردش کنترل‌های» خروجی «تصمیم»، عبارت‌هایی درون علامت [] نوشته می‌شود که نشانگر ادامه رویه در صورت برقرار بودن آن عبارت هستند. لازم است که در یک لحظه تنها یکی از مسیرها در پیش گرفته شوند. به عبارت دیگر، مسیرهای خروجی «تصمیم» نباید همپوشانی داشته باشند. برای مثال «تصمیم» دارای دو مسیر خروجی $[0 <= a]$ و $[a > 0]$ نادرست است، زیرا مساوی با صفر بودن در هر دو شرط مشترک است. برای تصحیح می‌توان شرط‌ها را به شکل $[a <= 0]$ و $[a > 0]$ تغییر داد.

۶-۲ - ادغام^۱

«ادغام»، یکی از عناصر کنترلی در نمودار فعالیت است که چند مسیر جایگزین را ترکیب می‌کند و ادامه روال را در قالب یک مسیر ادامه می‌دهد. این عنصر در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۰-۱۱: ادغام

این عنصر برای همگام‌سازی روال‌های موازی و همزمان استفاده نمی‌شود، یعنی هر چند که در «ادغام»، چندین ورودی وجود دارد، اما در هر اجرا، تنها یکی از ورودی‌ها به عنصر «ادغام» خواهد رسید (ورودی‌ها جایگزین هم هستند و نه موازی هم) و بعد از آن، رویه ادامه می‌یابد.

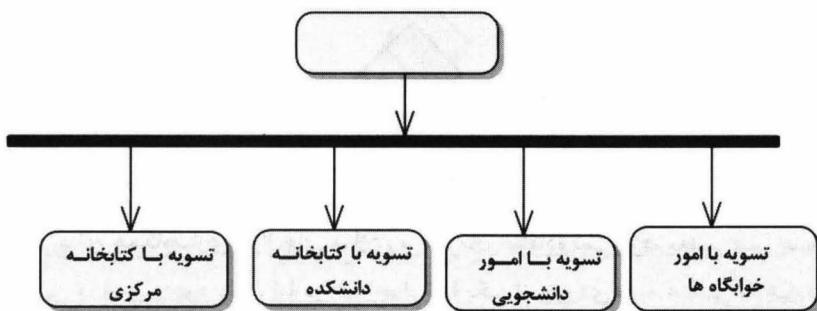
۲-۲- انشعباب^۱

«انشعاب» عنصری کترلی در نمودار فعالیت است که یک جریان ورودی را به چند جریان موازی و همروند تقسیم می‌کند. عنصر «انشعاب» یک ورودی و چندین خروجی دارد. این عنصر در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۱-۱۱: انشعباب عمودی و افقی

گاه در اجرای روال‌ها، چند فرایند به صورت همزمان آغاز می‌شوند و عملیات به صورت موازی ادامه می‌یابد. برای نشان دادن چنین شرایطی از عنصر «انشعاب» استفاده می‌شود. این عنصر، یک ورودی و چند خروجی دارد که خروجی‌ها نشان‌دهنده روال‌های موازی هستند. برای مثال در روال فارغ التحصیلی از دانشگاه، قبل از تأیید فارغ التحصیلی، باید روال تسویه با امور دانشجویی، کتابخانه، واحد آموزش و سایر واحدها انجام شود. تقدم و تأخیر انجام این فعالیت‌ها اهمیت ندارد و می‌توانند به صورت همزمان اجرا گردند.



شکل ۱۲-۱۱: نمونه انشعباب

عموماً «انشعاب» تنها یک ورودی دارد. از این‌رو در شرایطی که چند جریان ورودی، به شکل چند جریان موازی در «انشعاب» ادامه می‌یابند، باید قبل از ورود جریان‌های ورودی به «انشعاب»، آن‌ها را با استفاده از «ادغام»، با هم یکی کرد و سپس جریان خروجی «ادغام» را به «انشعاب» وارد کرد. توجه شود که خروجی‌های «انشعاب» دارای شرط نیستند.

۱-۸-۲- الحق

«الحق» عنصری کترلی است که چند جریان ورودی همزمان را با هم همگام کرده، تا اتمام همه جریان‌های ورودی منتظر مانده، سپس اجرا را در قالب یک خروجی ادامه می‌دهد. به عبارت دیگر پس از اتمام جریان‌های موازی، برای نشان دادن انتظار برای پایان تمام جریان‌های موازی و ادامه اجرا از «الحق» استفاده می‌شود.

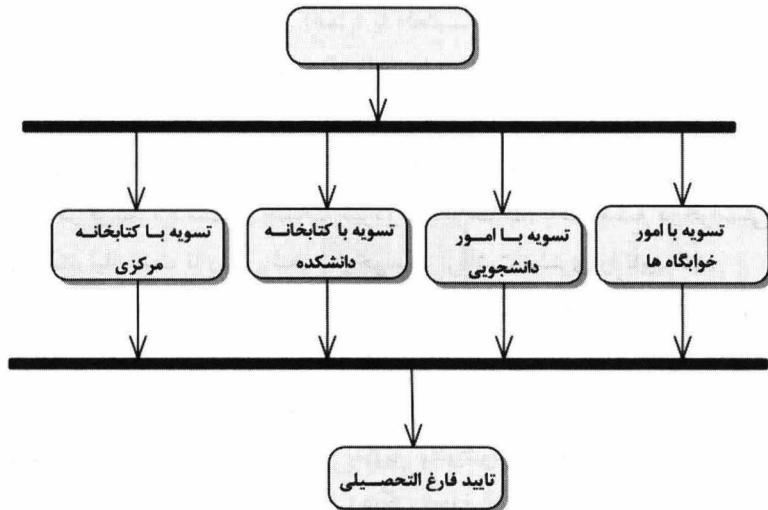
برخلاف «ادغام» که با رسیدن اولین ورودی اجرا را ادامه می‌دهد، «الحق» تا اتمام اجرای تمام جریان‌های ورودی منتظر مانده و پس از اتمام همه آن‌ها اجرا را ادامه می‌دهد.

نمایش این عنصر در UML مشابه عنصر «انشعاب» است، با این تفاوت که این عنصر چند ورودی داشته، دارای یک خروجی است. نمونه استفاده از این عنصر نیز در نمودار رسم شده برای عنصر «انشعاب» آمده است. این عنصر «همگام‌ساز»^۱ نیز نامیده می‌شود.

نمونه‌ای از «الحق» در شکل ۱۳-۱۱ نمایش داده شده است که تأیید فارغ‌التحصیلی پس از اتمام تمامی تسویه‌ها انجام می‌شود.

1. Join

2. Synchronizer

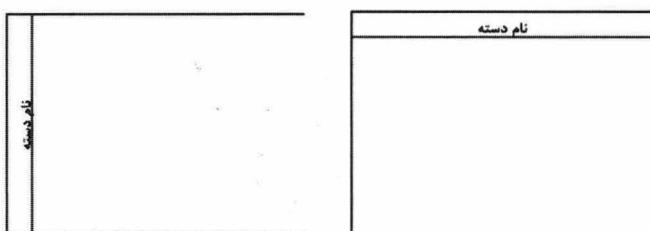


شکل ۱۱-۱۳: نمونه الحق

وجود دو عنصر «انشعاب» و «الحق» در نمودار فعالیت، بیانگر اجبار در اجرای موازی جریان‌ها نیست، بلکه نشان می‌دهد در صورت وجود قابلیت موازی‌سازی، امکان بهینه‌سازی عملکرد برنامه یا فرایند با استفاده از موازی‌سازی وجود دارد.

۹-۲- جداساز^۱

«جداساز» عنصری برای گروه‌بندی عناصر دارای خصوصیات مشترک در نمودار فعالیت است. این عنصر در UML به صورت زیر نمایش داده می‌شود.



شکل ۱۱-۱۴: جداساز عمودی و افقی

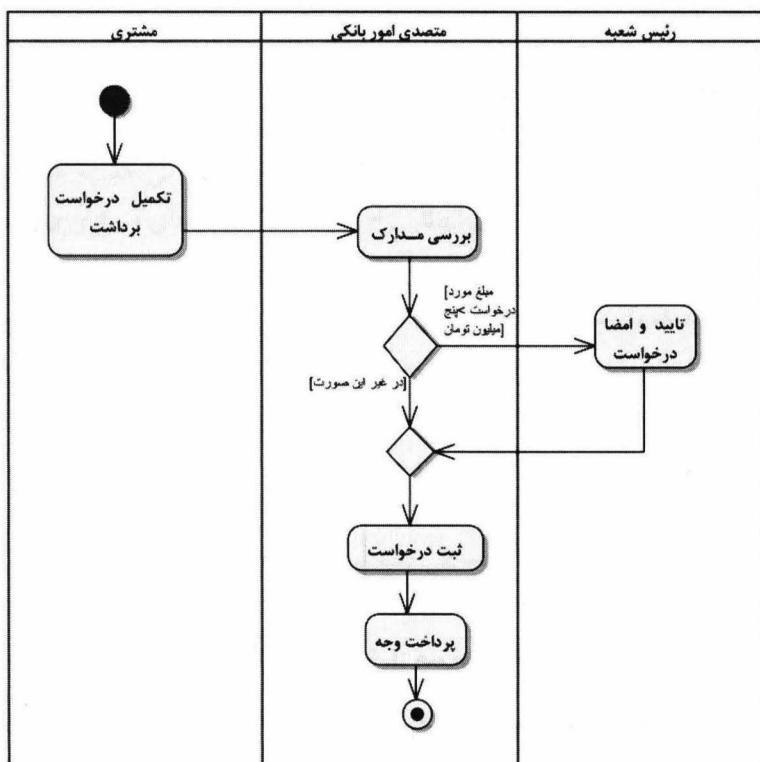
«جداساز» عناصر نمودار فعالیت را بر اساس خصوصیت یا موضوعی دسته‌بندی می‌کند. موضوع گروه‌بندی کننده می‌تواند موارد متعددی باشد. از جمله این موضوعات می‌توان به مجری گام‌های روال، واحد سازمانی مسئول، نوع «فعالیت» و «عمل» اشاره کرد.

در بسیاری از موارد از ویژگی مجری «عمل» یا «فعالیت» برای دسته‌بندی عناصر نمودار استفاده می‌شود. در این حالت، نام «جداساز» مشابه نام واحد یا پست سازمانی موجود در کسب‌وکار است. «جداساز»، نمودار فعالیت را به سطراها یا ستون‌هایی (بسته به جهت نمودار فعالیت) تقسیم می‌کند و هر سطر یا ستون شامل اقداماتی است که در یک گروه قرار می‌گیرند.

برای نمونه، در فرایند برداشت از حساب سپرده کوتاه مدت، اگر مبلغ درخواستی بیش از ۵ میلیون تومان باشد، نیاز است تا رئیس شعبه، درخواست برداشت مشتری را تأیید کند.

در انجام این فرایند، مشتری، متصلی امور بانکی و رئیس شعبه دخیل هستند که هر یک مجموعه‌ای از فعالیت‌ها را برای تکمیل فرایند انجام می‌دهد. برای نشان دادن فعالیت‌های هر یک افراد یا گروه‌ها از «جداساز» استفاده می‌شود (شکل ۱۵-۱۱).

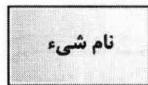
«جداساز» می‌تواند به صورت دو بعدی نیز نمایش داده شود تا نمودار فعالیت و گام‌های آن را به صورت ماتریسی $m \times n$ گروه‌بندی کند. از طرف دیگر، «جداساز» می‌تواند چند سطحی نیز باشد. در نسخه نخست UML، این عنصر تنها یک بعدی و ساده بود و به نام «خط شنا»^۱ خوانده می‌شد.



شکل ۱۵-۱۱: نمونه جداساز

۱۰-۲ - شیء^۱

«شیء» در نمودار فعالیت نشان‌دهنده نمونه‌ای از یک ساختار اطلاعاتی یا کلاس است که داده‌ای را در خود جای داده است و اطلاعات را بین دو «عمل» جابه‌جا می‌کند. این عنصر در شکل ۱۶-۱۱ نمایش داده شده است.

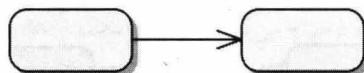


شکل ۱۶-۱۱: شیء

در نمایش «شیء» در UML نام «شیء» یا کلاس متناظر آن، درون آن نوشته می‌شود. «شیء» نشان‌گر ساختار اطلاعاتی است که در نقطه‌ای از فعالیت توسط «عمل‌ها» ایجاد، ویرایش یا استفاده می‌گردد.

۱۱-۲ - گردش شیء^۲

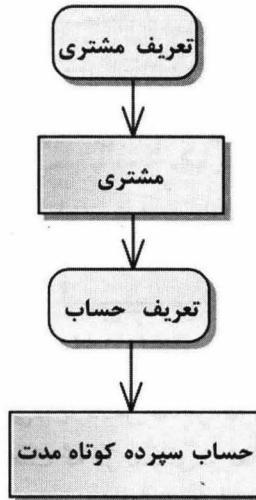
«گردش شیء» پیکان ارتباط‌دهنده بین دو عنصر در نمودار فعالیت است که انتقال اطلاعات بین آن‌ها را نشان می‌دهد. «گردش شیء» در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۷-۱۱: گردش شیء

برخی اوقات داده‌ها جنبه مهمی از رویه‌ای مدل‌شده هستند. برای نمایش حرکت داده‌ها در نمودار فعالیت از «گردش شیء» استفاده می‌گردد. وجود «گردش شیء» بین دو «عمل» در نمودار فعالیت به این معناست که «عمل‌ها» این داده را ایجاد کرده، تغییر داده یا از آن استفاده می‌نمایند. بدیهی است زمانی که «عمل» برای اجرا نیازمند شیئی باشد، تا زمان آماده‌شدن شیء، اجرا نخواهد شد.

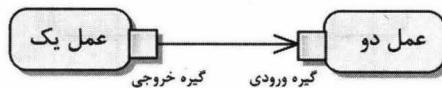
1. Object
2. Object flow



شکل ۱۸-۱۱: نمونه شیء و گردش شیء

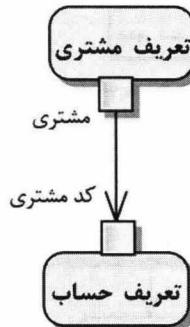
۱۲-۲ - گیره^۱

«گیره» عنصری است که نشان‌دهنده داده‌های ورودی به «عمل» یا خروجی از آن است. این عنصر در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۹-۱۱: گیره

چهارضلعی‌های کوچکی که در طرفین «عمل‌ها» کشیده شده‌اند، نشان‌دهنده «گیره‌های» ورودی و خروجی آن هستند. تا زمانی که تمام «گیره‌های» ورودی «عمل» آماده نباشند، آن «عمل» اجرا نخواهد شد. «گیره‌ها» شبیه پارامترهای ورودی و مقادیر خروجی در فرآخوانی «عمل» هستند. محدودیتی در نام‌گذاری «گیره» وجود ندارد. عموماً «گیره» به نام نوع اشیاء یا داده‌هایی که از طریق آن در جریان است، نام‌گذاری می‌شود.



شکل ۱۱-۲۰: نمونه گیره

در شکل قبلی، خروجی «تعريف مشتری»، «مشتری» است. از طرف دیگر، ورودی «تعريف حساب» نیز «کد مشتری» است.
 «پارامتر ممتد»^۱، «پارامتر خروجی استثناء»^۲، «مجموعه پارامترها»^۳ و «گیره ارزش»^۴ از انواع «گیره» هستند.

۱۳-۲ - انباره داده^۵

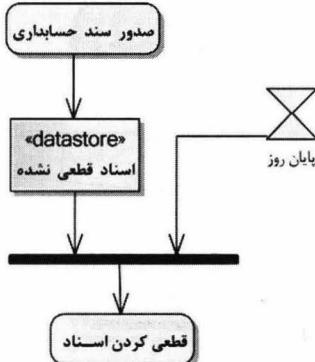
«انباره داده» عنصری برای نمایش محل نگهداری داده‌های پایدار و مانا^۶ در نمودار فعالیت است. «انباره داده» داده‌های ورودی را در خود ذخیره می‌کند و هنگام انتقال آن‌ها به مراحل دیگر، نسخه‌ای (کپی) از آن‌ها را ارسال می‌کند. هرگاه «شیئی» به «انباره داده» وارد شود که قبلاً در آن وجود داشته باشد، مقدار موجود «شیء» در «انباره داده»، با مقدار جدید جایگزین می‌شود. «انباره داده» در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۱-۲۱: انباره داده

انتخاب و تبدیل داده‌ها می‌تواند به لبه‌های خروجی «انباره داده» اضافه شود تا اطلاعات را از آن بازیابی کند.

1. Streaming parameter
2. Exception output parameter
3. Parameter set
4. Value pin
5. Data store
6. Persistence

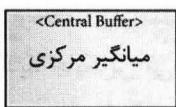


شکل ۱۱-۲۲: نمونه انباره داده

«انباره داده» در شکل قبل نشان می‌دهد که اسناد پس از صدور، نگهداری می‌گردند تا در پایان روز قطعی گردند.

۱۴-۲ - میانگیر مرکزی^۱

«میانگیر مرکزی» عنصری از جنس «شیء» برای مدیریت جریان‌های ورودی از چندین مبدأ و خروجی به چندین مقصد است که مبدأها و مقصد های مذکور به «عمل» یا «فعالیت» مشخصی متصل نیستند. «میانگیر مرکزی» در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۱-۲۳: میانگیر مرکزی

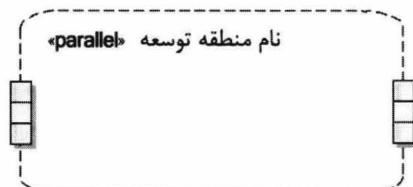
شکل زیر نشان می‌دهد که تولیدکنندگان یک و دو، محصولاتی را تولید می‌کنند که توسط مصرفکنندگان یک و دو استفاده می‌شوند. محصولات تولیدی در «میانگیر مرکزی» نگهداری می‌گردند و مصرفکنندگان محصولات مورد نیاز را در صورت وجود، از «میانگیر مرکزی» برمی‌دارند.



شکل ۱۱-۲۴: نمونه میانگیر مرکزی

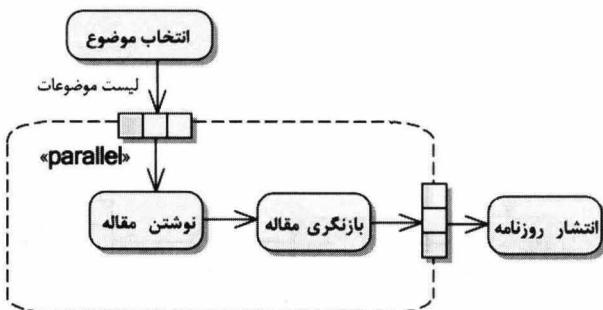
۱۰-۲ - منطقه توسعه^۱

«منطقه توسعه» نشان‌دهنده عنصری است که می‌تواند اجرای مجموعه‌ای از «عمل‌ها» را برای گروهی از ورودی‌ها تکرار کند. این عنصر به شکل زیر نمایش داده می‌شود.



شکل ۱۱-۲۵: منطقه توسعه

«منطقه توسعه» با یک مستطیل گوش‌گرد با خطوط خط‌چین و نواری شامل چند چهارضلعی در دو طرف آن مطابق شکل قبلی نمایش داده می‌شود. نوار چهارضلعی‌ها بیان‌گر مجموعه ورودی و خروجی هستند.



شکل ۱۱-۲۶: منطقه توسعه

شکل قبل نشان می‌دهد که ابتدا موضوعات روزنامه انتخاب می‌شوند، سپس مقاله هر موضوع نوشته می‌شود و بعد از آماده شدن، بازنگری می‌گردد. این کار برای مقاله‌ها به صورت موازی انجام می‌شود. پس از اتمام نوشتند و بازنگری همه مقاله‌ها، روزنامه منتشر می‌شود.

۱۶-۲ - پایان گردن^۲

«پایان گردن» عنصری است که بیان‌گر اتمام یکی از جریان‌های یک فعالیت است بدون آن که فعالیت اصلی به اتمام رسیده باشد. در نمودار فعالیت دو راه برای نمایش خاتمه فعالیت یا شاخه‌ای از فعالیت‌ها به شرح ذیل وجود دارد.

1. Expansion region
2. Flow final

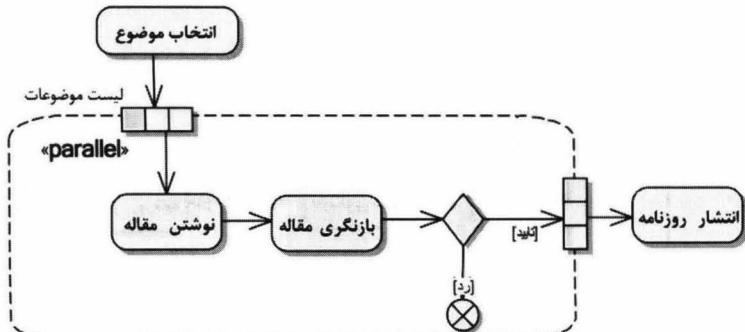
«نقطه پایان» بیانگر اتمام کل روال نمایش داده شده در نمودار است. به عنوان مثال در شکل قبلی، با ارسال خطاهای به تیم پیاده‌سازی، روال به اتمام خواهد رسید، در نتیجه از «نقطه پایان» استفاده می‌گردد.

«پایان گردش» برخلاف «نقطه پایان»، بیانگر به اتمام رسیدن روال نیست، بلکه نشان می‌دهد که یکی از جریان‌های موجود در روال به اتمام رسیده است، بدون آن که کل روال خاتمه یافته باشد. این عنصر به شکل زیر نمایش داده می‌شود.



شکل ۱۱-۲۷: پایان گردش

در شکل زیر که تغییر یافته شکل ۱۱-۲۶ است، در صورت رد مقاله در بازنگری، مقاله برای انتشار ارسال نمی‌گردد. به عبارت دیگر تنها جریان مرتبط با مقاله رد شده به پایان می‌رسد و فعالیت و رویه انتشار برای سایر مقاله‌ها ادامه می‌یابد.



شکل ۱۱-۲۸: نمونه پایان گردش

۱۷-۲ - رخداد زمانی^۱

«رخداد زمانی» عنصری برای نشان دادن رخدادهای وابسته به زمان در نمودار فعالیت است. این عنصر به شکل زیر نمایش داده می‌شود.



شکل ۱۱-۲۹: رخداد زمانی

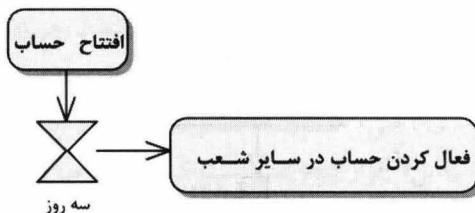
گاهی زمان عامل مهمی در اجرای فعالیت است. رخداد زمانی «سه روز انتظار بعد از افتتاح حساب برای فعال شدن آن در تمامی شب» یا فعالیت تکرارشونده در یک بازه زمان معین نظیر «محاسبه سود سپرده در پایان هر ماه» نمونه هایی از این دست هستند. برای مدل سازی وقایع زمانی از «رخداد زمانی» استفاده می شود.

از «رخداد زمانی» می توان برای موارد زیر استفاده نمود:

- زمان انتظار تا آغاز یک رخداد

فعالیت پس از «رخداد زمانی»، باید مدتی در انتظار باشد تا اجرا شود. در این حالت، نوشته کنار «رخداد زمانی» نشان دهنده مدت زمان انتظار است.

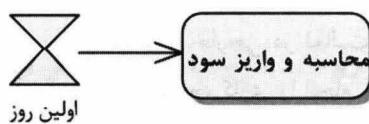
در شکل زیر نشان داده شده است که پس از افتتاح حساب در شعبه افتتاح گشته، سه روز طول خواهد کشید تا حساب وی در سایر شب های نیز قابل شناسایی و خدمت رسانی باشد.



شکل ۱۱-۳۰: نمونه رخداد زمانی برای نمایش زمان انتظار

- رخداد تکرار پذیر در بازه های زمانی

از «رخداد زمانی» می توان برای نمایش رخداد قابل تکرار در بازه های زمانی نیز استفاده نمود. برای مثال در سیستم «حساب سپرده کوتاه مدت» در اولین روز هر ماه، سود ماه گذشته حساب های مشتریان محاسبه و واریز می گردد.

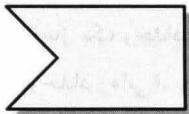


شکل ۱۱-۳۱: نمونه رخداد زمانی برای نمایش رخداد تکرار پذیر

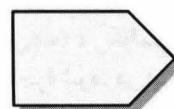
«رخداد زمانی» جایگزینی برای شروع یک فعالیت است. در نمودار فعالیت ممکن است عنصری برای نمایش آغاز به کار فعالیت وجود نداشته باشد و فعالیت بر اثر «رخداد زمانی» آغاز گردد.

۱۸-۲ - سیگنال^۱

«سیگنال» عنصری است که امکان نمایش تعامل با اجزای خارج از سیستم را فراهم می‌کند. این ارتباط می‌تواند دریافت پیام از اجزای خارجی یا ارسال پیام به آن‌ها باشد. «سیگنال ارسال» و «سیگنال دریافت»^۲ پیام دارای دو نماد متفاوت به شکل زیر هستند.

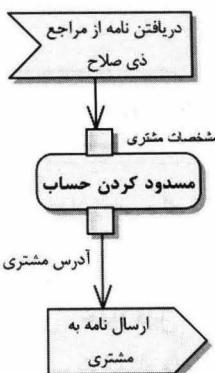


شکل ۱۱-۳۳: سیگنال دریافت



شکل ۱۱-۳۲: سیگنال ارسال

به عنوان مثال، به محض دریافت نامه از مراجع ذی‌صلاح برای مسدودی حساب، ابتدا حساب مشتری مسدود می‌گردد و پس از آن نامه‌ای جهت آگاهی به وی ارسال می‌گردد. این رویه در شکل زیر نشان داده شده است.



شکل ۱۱-۳۴: سیگنال دریافت

«سیگنال ارسال» پیامی است که به شرکت‌کننده خارجی در فعالیت ارسال می‌شود. وقتی فرد یا سیستم خارجی پیام را دریافت کرد، احتمالاً در پاسخ، کاری را انجام خواهد داد که در نمودار فعالیت مدل نخواهد شد. ارسال «سیگنال» به صورت ناهمگام^۳ انجام می‌گردد یعنی فعالیت ارسال کننده، متظر پاسخ گیرنده «سیگنال» نمی‌ماند و بلاضافله به «عمل» بعدی می‌رود. دریافت «سیگنال» نیز موجب اجرای «عمل» در نمودار فعالیت می‌گردد. گیرنده «سیگنال» می‌داند که «سیگنال» خواهد رسید و حتی می‌داند چگونه به آن واکنش نشان دهد، ولی زمان دقیق آن را نمی‌داند.

1. Signal
2. Send signal
3. Receive signal
4. Asynchronous

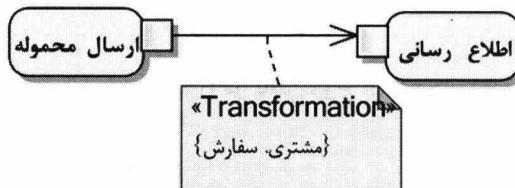
گاه «عمل» در انتظار رخدادی خاص یا دریافت «سیگنال» است. برای این کار می‌توان از «سیگنال» دریافتی استفاده کرد که دارای «گرددش کترل» ورودی نیست. نبود «گرددش کترل» ورودی به معنای انتظار دائم برای دریافت «سیگنال» در زمان اجرای فعالیت است.

اگر فعالیت جاری «سیگنالی» را ارسال می‌کند و منتظر دریافت پاسخ آن می‌ماند، برای نمایش انتظار دریافت پاسخ از «سیگنال» دریافتی استفاده می‌گردد.

۱۹-۲ - نشانه^۱

(نشانه)، «جريان داده» یا «جريان کترل» را در نمودار فعالیت مدیریت می‌کند. «نشانه» می‌تواند شامل یک شیء، بخشی از داده یا شرط کترلی باشد. به عبارت دیگر، «نشانه کترلی» یا «نشانه شیئی» از گامی به گام دیگر در نمودار فعالیت در حرکت هستند و حرکت آن‌هاست که اجرای نمودار را به همراه دارد.

«گرددش شیء» یا «گرددش کترل» می‌توانند قاعده‌هایی برای کترول جريان حرکت و جريان «نشانه» مشخص کنند. به عنوان مثال در شکل زیر، خروجی «عمل» «ارسال سفارش»، شیء سفارش است. ورودی «عمل» «اعلام به مشتری»، شیء مشتری است. برای استخراج اطلاعات مشتری از اطلاعات سفارش، قاعده‌ای به شکل «تبديل»^۲ قرار داده شده است که نشان می‌دهد، حوزه اطلاعاتی (فیلد) مشتری در شیء سفارش به «عمل» بعدی منتقل می‌گردد.



شکل ۱۱-۳۵: تبدیل گرددش شیء

از آنجا که بسیاری از نکات مربوط به «نشانه» در سایر عناصر توضیح داده شده است، از بیان مجدد آن‌ها در این قسمت خودداری شده است.

۲۰-۲ - بست^۳

«بست» عنصری برای ساده‌سازی نمایش نمودار فعالیت است. این عنصر به شکل زیر نشان داده می‌شود.

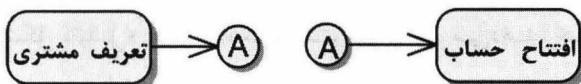
1. Token
2. Transformation
3. Connector



شکل ۱۱-۳۶: بست

این عنصر معنا یا منطق خاصی را به نمودار نمی‌افزاید و تنها برای ساده‌سازی نمودارهای پیچیده یا کاهش پیچیدگی ارتباطات حاصل از «گردش کترل» و «گردش شیء» به کار می‌رود. اگر نمودار فعالیت دارای تعداد زیادی «عمل» باشد، معمولاً با نموداری شلوغ، طولانی و خطوطی متقطع روبرو خواهد شد که خواندن آن با دشواری همراه خواهد بود. برای کاهش این گونه پیچیدگی، از «بست» استفاده می‌شود. «بست» با یک دایره با نامی داخل آن ترسیم می‌شود. معمولاً به «بست» نام تک حرفی داده می‌شود.

در نمودار فعالیت، «بست» به صورت جفت‌های دوتایی ظاهر می‌شود که یکی جریانی ورودی و دیگری جریانی خروجی دارد. «بست» دوم فرایند را از جایی ادامه می‌دهد که «بست» اول آن را قطع کرده است. شکل زیر نشان‌دهنده استفاده از «بست» در یک نمودار فعالیت است.

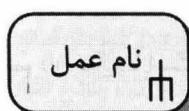


شکل ۱۱-۳۷: نمونه بست

استفاده زیاد از «بست» در نمودار، بیننده را در یافتن زوج آن‌ها دچار زحمت خواهد کرد.

۲۱-۲ مدیریت نمودارهای پیچیده

گاه نمودار فعالیت بسیار بزرگ و پیچیده می‌گردد. در این گونه نمودارها معمولاً توالی مشخصی از «عمل‌ها» بیش از یکبار رخ می‌دهند. در این شرایط، می‌توان با انتقال توالی تکراری به نمودار دیگری، خوانایی را افزایش و پیچیدگی نمودار اصلی را کاهش داد. برای نمایش فراخوانی نمودار فعالیت جدید در نمودار فعالیت اصلی از «عمل»‌ای به شکل زیر استفاده می‌شود.



شکل ۱۱-۳۸: عمل فراخواننده نمودار فعالیت دیگر

در این نوع «عمل»، علامت چنگک مانندی به نماد «عمل» معمولی افزوده می‌شود. این «عمل»

معمولًا با نام نمودار فعالیتی که فراخوانی می‌شود، همنام است. این کار مانند فراخوانی یک متده برنامه‌نویسی در درون متده دیگری است.

۳- نکات کلیدی

- نمودار فعالیت یکی از پرکاربردترین ابزارها در تحلیل سیستم است.
- نمودار فعالیت می‌تواند برای نمایش فرایندهای کسب و کار، جزئیات مورد کاربرد و الگوریتم‌ها به کار رود.
- «فعالیت» و «عمل» غالباً به اشتباه به جای هم به کار می‌روند. «عمل» مانند دستورات زبان برنامه‌نویسی و «فعالیت» مانند متدها یا توابع هستند.

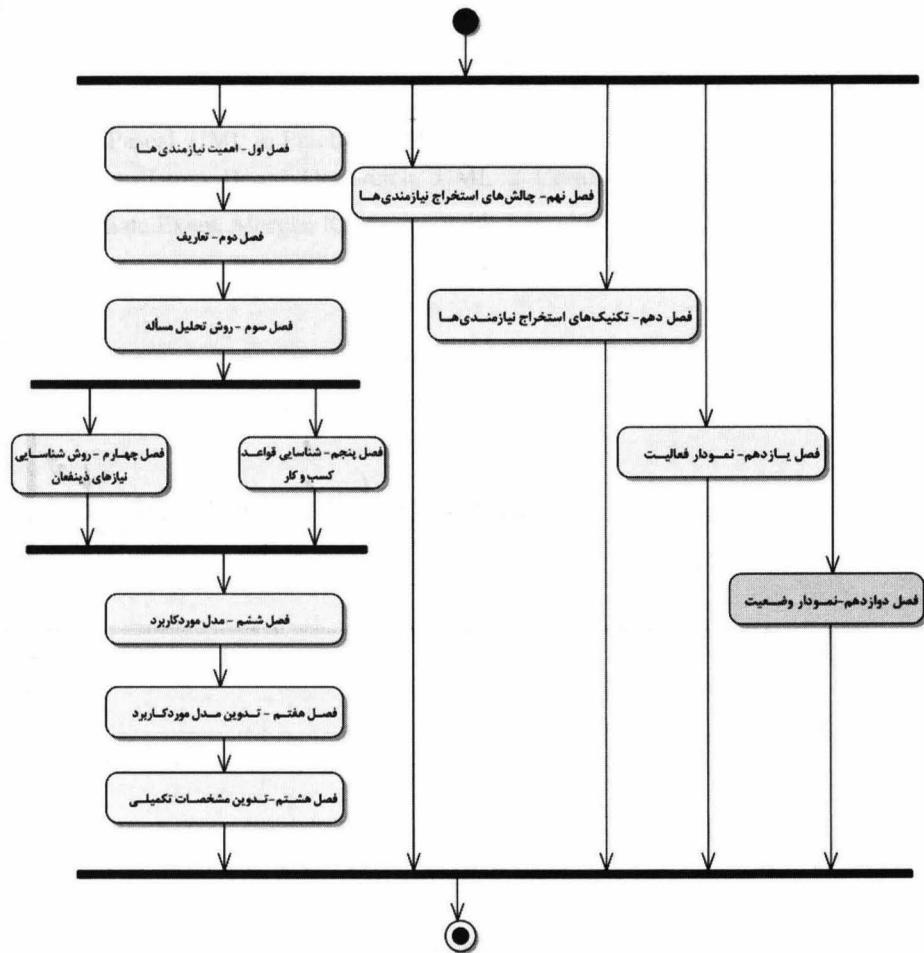
۴- مراجع

1. Hamilton, Kim و Russell Miles. Learning UML 2.0. O'Reilly, 2006
2. Pender, Tom. UML Bible. John Wiley & Sons, 2003
3. Roques, Pascal. UML in Practice. Wiely, 2004
4. Weilkiens, Tim و Bernd Oestereich. UML 2 Certification Guide: Fundamental & Intermediate Exam. Morgan Kaufmann Publishers, 2006

فصل دوازدهم - نمودار و ضعیت

«مهم ترین مسئله در یک رابطه، شنیدن چیزهایی است که گفته نمی شود.»

پیتر دراکر



۱- مقدمه

نمودار وضعیت یکی از ابزارهای قوی در تحلیل سیستم‌ها است. این نمودار برای نمایش وضعیت‌ها و تغییر آن‌ها به کار گرفته می‌شود. وضعیت‌ها می‌توانند متعلق به یک کلاس، مورد کاربرد، سیستم یا زیرسیستم یا هر موضوع دیگری باشند.

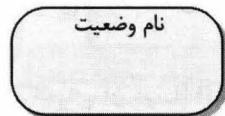
این فصل به تشریح نمودار وضعیت و اجزای آن در UML می‌پردازد. اجزایی از نمودار وضعیت که در این فصل مورد بررسی قرار می‌گیرند، عبارتند از:

- وضعیت^۱
- گذار^۲
- وضعیت مرکب^۳
- وضعیت پایانی^۴
- شبه وضعیت^۵
- وضعیت آغازین^۶
- نقطه ورود^۷
- نقطه خروج^۸
- انتخاب^۹
- پیوند^{۱۰}
- تاریخچه سطحی^{۱۱}
- تاریخچه عمیق^{۱۲}
- پایان گر^{۱۳}
- انشعاب^{۱۴}
- الحق^{۱۵}

-
1. State
 2. Transition
 3. Composite state
 4. Final state
 5. Pseudo state
 6. Initial state
 7. Entry point
 8. Exit point
 9. Choice
 10. Junction
 11. Shallow history
 12. Deep history
 13. Terminate
 14. Fork
 15. Join

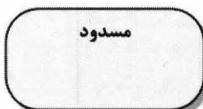
۲- وضعیت^۱

«وضعیت» نشان‌دهنده موقعیتی است که در آن شرایط معینی برقرار می‌گردد. این عنصر در UML به شکل زیر نمایش داده می‌شود.



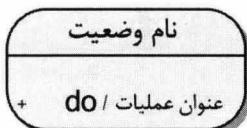
شکل ۱-۱۲: وضعیت

به عنوان مثال در شکل زیر یکی از «وضعیت‌های» شیء حساب به نام «مسدود» نشان داده شده است. «وضعیت» مسدود به شرایطی اشاره دارد که در آن حساب به دلیل دستور مراجعت ذی‌صلاح مسدود شده است و امکان انجام تراکنش‌های بانکی مانند واریز به حساب و برداشت از آن وجود ندارد.



شکل ۱-۱۲-۲: وضعیت

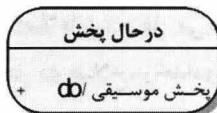
موقعیتی که «وضعیت» نشان‌دهنده آن است، می‌تواند ایستا یا پویا باشد. اگر «وضعیت»، نشان‌دهنده شرایطی پویا در سیستم باشد، عملیاتی را که شیء در آن «وضعیت» انجام می‌دهد، «رفتار داخلی»^۲ نامیده و به شکل زیر نشان داده می‌شود.



شکل ۱-۱۲-۳: رفتار داخلی

برای مثال، فرض کنید دستگاه پخش‌کننده لوح‌فشرده^۳ موسیقی دارای «وضعیت‌های» روشن، درحال پخش و خاموش باشد. با ورود به «وضعیت» در حال پخش، انتظار می‌رود که پخش لوح‌فشرده شروع گردد و تا زمان خروج از این «وضعیت» ادامه پیدا کند. این موضوع به شکل زیر مدل می‌گردد.

1. State
2. Internal behaviour
3. CD

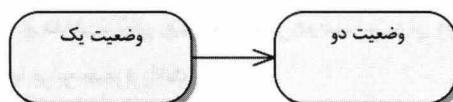


شکل ۱۲-۴: نمونه رفتار داخلی

دو نوع خاص از «رفتار داخلی» وجود دارد که «رفتار ورودی»^۱ و «رفتار خروجی»^۲ نامیده می‌شوند که برای نشان دادن آن‌ها به جای برجسب انجام (do)، از برجسب‌های ورودی (entry) و خروجی (exit) استفاده می‌شود. «رفتار ورودی» نشان‌دهنده فعالیتی است که به محض ورود به «وضعیت» و فعال شدن آن انجام می‌شود و «رفتار خروجی» نشان‌دهنده فعالیتی است که در هنگام خروج از یک «وضعیت» انجام می‌شود. اجرای این دو رفتار، بی‌وققه^۳ انجام می‌شود، یعنی توقفی در حین اجرا رخ نمی‌دهد.

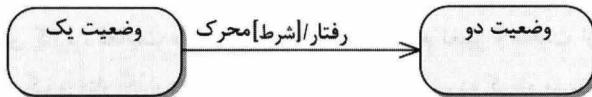
۳- گذار^۴

یال ارتباط‌دهنده دو وضعیت، «گذار» نامیده و در UML به شکل زیر نشان داده می‌شود.



شکل ۱۲-۵: گذار

در شکل بالا پیکان ارتباط‌دهنده وضعیت یک به وضعیت دو، یک «گذار» است. درواقع «گذار» نشان‌گر تغییر حالت از یک «وضعیت» به «وضعیت» دیگر است. رویدادی که موجب تغییر وضعیت می‌شود، بر روی «گذار» نوشته می‌شود. شیوه‌نامه نمایش رویداد به صورت «رفتار/[شرط] محرک» است که این اجزا در ادامه تشریح می‌شوند.



شکل ۱۲-۶: محرک، شرط و رفتار در گذار

۱-۳- محرک

«محرك» معرف رخدادی است که منجر به تغییر «وضعیت» می‌شود. برای مثال در سیستمی که ورودی‌های کاربر را پردازش می‌کند، رخداد «افشردگی شدن کلید صفحه کلید»، سیستم را از وضعیت

1. Entry behaviour
2. Exit behaviour
3. Uninterruptable
4. Transition

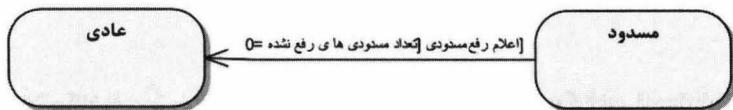
«دریافت اطلاعات» به وضعیت «پردازش اطلاعات» منتقل می‌کند. در سیستم حساب سپرده کوتاه مدت نیز اعلام مسدودی حساب از طرف مراجع ذی صلاح، رخدادی است که منجر به مسدودی حساب می‌گردد که به شکل «محرك» مدل می‌گردد.



شکل ۱۲-۷: مدل‌سازی اعلام مسدودی حساب به شکل محرك

۲-۳- شرط

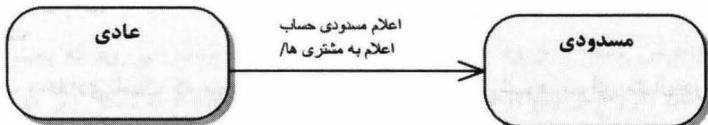
با وقوع رخداد («محرك»)، در صورتی که «شرط» بیان شده درست باشد، انتقال از وضعیت انجام می‌گردد. به عنوان مثال، در سیستم پردازشگر ورودی‌های کاربر، در صورتی «گذار» بین وضعیت «دریافت اطلاعات» و «پردازش اطلاعات» انجام می‌گردد که شرط «تعداد ورودی‌ها بیش از ۱۰ نویسه^۱» درست باشد. در صورتی که تعداد نویسه‌ها کمتر از ۱۰ باشد، تغییر «وضعیت» انجام نمی‌شود. در سیستم حساب سپرده کوتاه مدت نیز در صورتی تغییر «وضعیت» از «مسدود» به «عادی» انجام می‌شود که «تعداد مسدودی‌ها برابر صفر» باشد.



شکل ۱۲-۸: نمونه شرط

۳-۳- رفتار

«رفتار» یا «رفتار گذار» فعالیتی است که در حین تغییر «وضعیت» انجام می‌شود. برای مثال در سیستم پردازشگر ورودی‌های کاربر، فعالیت «ارسال اطلاعات» در هنگام تغییر وضعیت از «ورود اطلاعات» به «پردازش اطلاعات»، یک «رفتار گذار» است. در سیستم حساب سپرده کوتاه مدت نیز، اعلام به مشتری در هنگام انتقال «وضعیت» از عادی به مسدود یک «رفتار» است.



شکل ۱۲-۹: نمونه رفتار

«گذار» بر اساس منطقی که ترکیبی از «محرك» و «شرط» است، اجرا می‌شود. این منطق در جدول زیر تشریح شده است.

جدول ۱۲-۱: اجرای گذار بر اساس محرك و شرط

کاربرد	شرط انجام «گذار»	«محرك» دارد؟	«شرط» دارد؟
این حالت برای نمایش جلوگیری از تغییر وضعیت در صورت نادرستی یک شرط با وجود وقوع رخداد مورد انتظار به کار می‌رود.	- رخدادن «محرك» - درستی «شرط»	بلی	بلی
این حالت برای نشان‌دادن تغییر وضعیت بر اثر بروز یک رخداد به کار می‌رود.	- رخدادن «محرك»	بلی	خیر
برای نشان‌دادن جلوگیری از اجرای یک «گذار» یا انتخاب یکی از بین چند «گذار» کاربرد دارد. بین چند «گذار» «گذاری» انجام می‌گردد که «شرط» آن درست باشد.	- درستی «شرط»	خیر	بلی
این حالت برای نشان‌دادن انجام «گذار» به محض اتمام «رفتار داخلی» «وضعیت» به کار می‌رود.	ندارد	خیر	خیر

در صورتی که «وضعیت» مبدأ و مقصد در «گذار» یکی باشد، آن را «خودگذار»^۱ می‌نامند. در شکل زیر نمونه‌ای از این «گذار» نشان داده شده است.

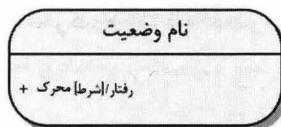


شکل ۱۰-۱۲: خودگذار

۴-۳- گذار داخلی

نوع خاصی از «گذار» که «گذار داخلی» نامیده می‌شود، منجر به واکنشی درون «وضعیت» می‌گردد، اما منجر به تغییر «وضعیت» نمی‌شود. برخلاف «خودگذار» که منجر به رخدادن «رفتار ورودی» و «رفتار خروجی» می‌گردد، این نوع «گذار» منجر به آن‌ها نمی‌گردد. این نوع «گذار» برای نشان‌دادن رخدادهایی که منجر به تغییر وضعیت شیء نمی‌گردد، به کار می‌رود.

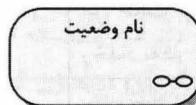
نمایش این نوع «گذار» در UML با شیوه‌نامه‌ای به شکل «رفتار/[شرط] محرك» در درون «وضعیت» انجام می‌شود.



شکل ۱۱-۱۲: گذار داخلی

۴- وضعیت مرکب^۱

«وضعیتی» که خود شامل مجموعه‌ای از «وضعیت‌ها» در قالب یک یا چند نمودار وضعیت باشد، «وضعیت مرکب» نامیده می‌شود.

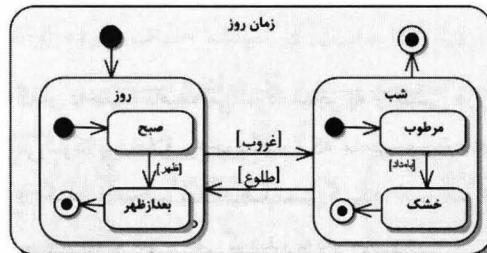


شکل ۱۲-۱۲: وضعیت مرکب

هر نمودار وضعیت در «وضعیت مرکب» در قسمتی به نام «منطقه»^۲ قرار می‌گیرد که به وسیله نقطه‌چین از سایر «منطقه‌ها» جدا می‌شود. هر یک از «وضعیت‌های» موجود در «وضعیت مرکب»، «زیروضعیت»^۳ نامیده می‌شود.

در نمودار وضعیت امکان نمایش وضعیت‌های همزمان یک شیء وجود دارد. برای نشان دادن وضعیت‌های همزمان از «وضعیت مرکب» استفاده می‌گردد. زمانی که «وضعیت مرکبی» فعال می‌شود، هریک از نمودارهای وضعیت درون آن شروع به اجرا می‌کنند. اگر «محركی» بر روی «وضعیت مرکب» رخ دهد، اجرای نمودارهای وضعیت درون آن متوقف می‌شود و «محرك» مذکور اجرا می‌گردد. اجرای «وضعیت مرکب» با اتمام فعالیت‌های «زیروضعیت‌های» درون آن پایان می‌یابد.

به عنوان مثال، «وضعیت مرکب» شکل ۱۲-۱۳ نمایان گر وضعیت یک شبانه‌روز است.



شکل ۱۲-۱۳: وضعیت مرکب

1. Composite state
2. Region
3. Substate

۵- وضعیت پایانی^۱

«وضعیت پایانی» اتمام نمودار وضعیت را نمایش می‌دهد و در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۴-۱۲: وضعیت پایانی

«وضعیت پایانی» دارای «رفتار داخلی»(entry, exit, do) و «گذار خروجی» نیست.

۶- شبه وضعیت^۲

«شبه وضعیت» به مجموعه‌ای از عناصر در نمودار وضعیت گفته می‌شود که برای ایجاد ارتباطات پیچیده بین «وضعیت‌ها» تعبیه شده‌اند. در UML یازده نوع از این عناصر وجود دارند که عبارتند از «وضعیت آغازین»، «نقطه ورود»، «نقطه خروج»، «انتخاب»، «پیوند»، «تاریخچه سطحی»، «تاریخچه عمیق»، «پایان‌گر»، «وضعیت پایانی»، «انشاء» و «الحق». «شبه وضعیت‌ها» در ادامه تشریح می‌گردند.

۶-۱- وضعیت آغازین^۳

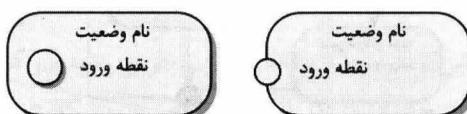
«وضعیت آغازین» شروع یک نمودار وضعیت را نمایش می‌دهد و در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۴-۱۳: وضعیت آغازین

۶-۲- نقطه ورود^۴

«نقطه ورود» نقطه ورود به یک «وضعیت مرکب» یا «ماشین وضعیت» را مشخص می‌کند. این عنصر در UML به شکل زیر نمایش داده می‌شود.

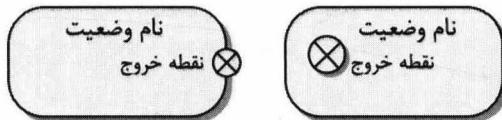


شکل ۱۴-۱۴: دو نمایش از نقطه ورود

1. Final state
2. Pseudo state
3. Initial state
4. Entry point
5. State machine

۳-۶- نقطه خروج^۱

«نقطه خروج» نقطه خروج از یک «وضعیت مرکب» یا «ماشین وضعیت» را مشخص می‌کند. این عنصر در UML به شکل زیر نشان داده می‌شود.



شکل ۱۷-۱۲: دو نمایش از نقطه خروج

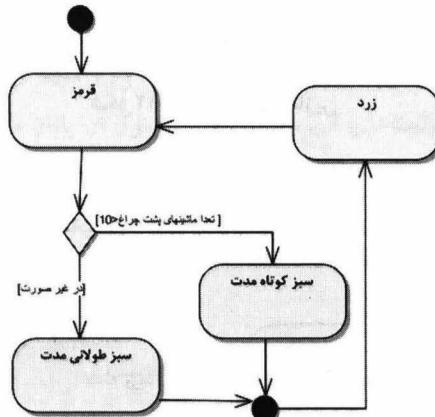
۴-۴- انتخاب^۲

«انتخاب» برای انتخاب یکی از چند «گذار» خروجی، بر اساس «شرطها» و «محرك‌های» هر «گذار» به کار می‌رود. این عنصر مشابه عنصر «تصمیم» در نمودار فعالیت است. این عنصر در UML به شکل زیر نمایش داده می‌شود.



شکل ۱۸-۱۲: انتخاب

شکل زیر نشان‌دهنده وضعیت‌های چراغ راهنمایی است. در این شکل، خروج از وضعیت قرمز با انتخاب یکی از دو وضعیت سبز کوتاه‌مدت یا سبز طولانی‌مدت بر اساس تعداد خودروها همراه است.



شکل ۱۹-۱۲: انتخاب

1. Exit point
2. Choice

۶-۵- پیوند^۱

این عنصر برای ترکیب چندین «گذار» مورد استفاده قرار می‌گیرد و در UML به صورت شکل ۲۰-۱۲ نمایش داده می‌شود. این عنصر شبیه «ادغام» در نمودار فعالیت است.



شکل ۲۰-۱۲: پیوند

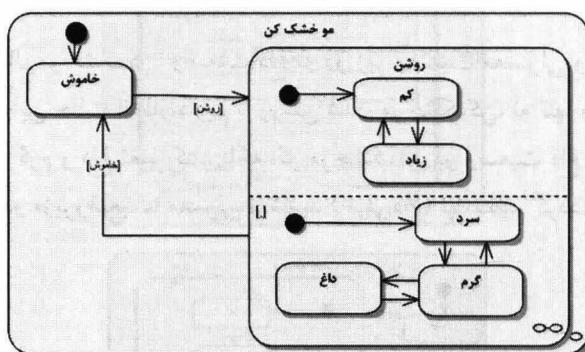
۶-۶- تاریخچه سطحی^۲

«تاریخچه سطحی»، آخرین «زیروضعیت» از «وضعیت» را قبل از خروج از آن، به خاطر می‌سپارد. این عنصر به شکل زیر نمایش داده می‌شود.



شکل ۲۱-۱۲: تاریخچه سطحی

به عنوان مثال یک موخشک‌کن (سشوار) را در نظر بگیرید که وضعیت‌های آن مانند شکل زیر است.

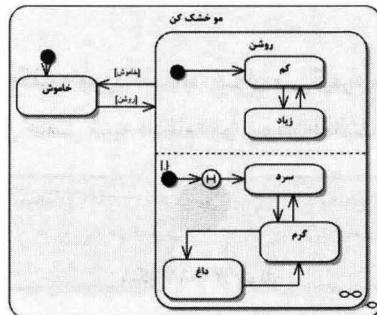


شکل ۲۲-۱۲: نمودار وضعیت موخشک کن

انتظار داریم پس از روشن کردن مجدد موخشک‌کن، آخرین وضعیت گرمایی (سرد، گرم و داغ) مطابق آخرین وضعیت قبل از خاموش کردن باشد. این موضوع به شکل زیر نشان داده می‌شود که در آن با استفاده از «تاریخچه سطحی»، آخرین وضعیت گرمایی قبل از خروج از وضعیت «روشن» ثبت می‌گردد تا در روشن شدن مجدد، موخشک‌کن با آن وضعیت آغاز به کار کند.

1. Junction

2. Shallow history



شکل ۱۲-۲۳: نمودار وضعیت موخشک کن با تاریخچه سطحی

هر «وضعیت مرکب می‌تواند حداکثر یک «تاریخچه سطحی» داشته باشد.

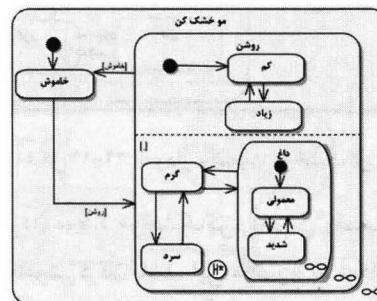
۶-۷- تاریخچه عمیق^۱

این عنصر، آخرین شرایط یک «زیروضعیت» فعال را در خود نگه می‌دارد. «تاریخچه عمیق» به شکل زیر در UML نشان داده می‌شود.



شکل ۱۲-۲۴: تاریخچه عمیق

فرض کنید در مثال موخشک کن، وضعیت داغ به دو زیر وضعیت معمولی و شدید (خیلی داغ) تقسیم شده باشد. در این حالت انتظار داریم با روشن شدن موخشک کن، نه تنها وضعیت به آخرین وضعیت از بین سرد، گرم و داغ تغییر کند، بلکه اگر موخشک کن در وضعیت داغ بود (یک «وضعیت مرکب»)، به یکی از دو «زیروضعیت» معمولی و شدید (خیلی داغ) نیز تنظیم گردد.



شکل ۱۲-۲۵: تاریخچه عمیق

در هر «وضعیت مرکب» حداکثر یک «تاریخچه عمیق» وجود دارد.

۶-۸- پایان گر^۱

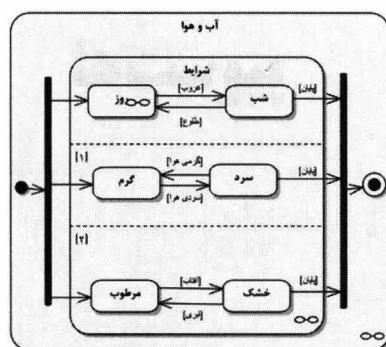
«پایان گر» پایان یافتن اجرای یک «ماشین وضعیت» را در نمودار وضعیت نشان می‌دهد. در UML از نماد زیر برای نشان دادن این عنصر استفاده می‌شود.



شکل ۱۲-۲۶: پایان گر

۶-۹- انشعباب^۲ و الحق^۳

این عنصر مانند «انشعاب» در نمودار فعالیت است با این تفاوت که ورودی و خروجی‌های آن، «گذار» هستند. «وضعیت‌هایی» که «گذارهای» «انشعاب» به آن‌ها وارد می‌شوند، هم‌مان فعال می‌شوند. «الحق» نیز مشابه عنصر «الحق» در نمودار فعالیت است. چندین «گذار» به «الحق» وارد شده و تنها یک «گذار» از آن خارج می‌گردد. در شکل زیر کاربردی از این دو عنصر نشان داده شده است.



شکل ۱۲-۲۷: انشعباب و الحق

۷- نکات کلیدی

- نمودار وضعیت یکی از ابزارهای مهم تحلیل است.
- این نمودار برای نمایش وضعیت‌ها و تغییر آن‌ها به کار گرفته می‌شود.
- وضعیت‌ها می‌توانند متعلق به یک کلاس، موردنکاربرد، سیستم یا زیر سیستم یا هر موضوع دیگری باشند.

1. Terminate
2. Fork
3. Join

۸- مراجع

1. Hamilton, Kim, Russell Miles. Learning UML 2.0. O'Reilly, 2006
2. Pender, Tom. UML Bible. John Wiley & Sons, 2003
3. Roques, Pascal. UML in Practice. Wiely, 2004
4. Weilkiens, Tim, Bernd Oestereich. UML 2 Certification Guide: Fundamental & Intermediate Exam. Morgan Kaufmann Publishers, 2006



پیوست‌ها

پیوست یک – واژه‌نامه

پیوست دو – خلاصه انگلیسی

پوستیک - واژه نامه



۱- واژه‌نامه فارسی به انگلیسی

Test case	آزمایه
Test	آزمون
"Yes, But" syndrome	آفت «بله، اما»
The "User and the Developer" syndrome	آفت کاربر - توسعه‌دهنده
Authentication	احراز هویت
Merge	ادغام
Requirements elicitation	استخراج نیازمندی‌ها
Deployment	استقرار
Data store	انباره داده
Choice	انتخاب
Fork	انشعاب
Quality goals	اهداف کیفی
Development priority	اولویت توسعه
Role playing	ایفای نقش
Connector	بست
Flow final	پایان گردش
Terminate	پایان گر
Scanner	پویش گر
Junction	پیوند
Shallow history	تاریخچه سطحی
Deep history	تاریخچه عمیق
Survey/Questionarie	تحقيق/پرسشنامه
Analysis	تحلیل
Problem analysis	تحلیل مسائل
Documents analysis	تحلیل مستندات
Precedence and priority	ترتیب و اولویت‌بندی

Decision	تصمیم
Repairability	تعمیرپذیری
Generalization	تعمیم
Throughput	توان عملیاتی
Extendibility	توسعه‌پذیری
Brainstorming	توفان ذهنی
Product positioning	جایگاه محصول
Partition	جداساز
Partition	جداساز
Lifecycle	چرخه زندگی
Vision	چشم انداز
Join	الحق
Copyright	حق تأليف
Domain expert	خبره حوزه مسئله
Subject matter expert	خبره موضوع
System attributes	خصوصیات سیستم
Quality attributes	خصوصیات کیفی
Policy	خط مشی
Usability	خوشکاری
RFP- Request for proposal	درخواست برای پیشنهاد
Availability	دسترس‌پذیری
Stakeholder	ذینفع
Interface	رابط
Application programming interface (API)	رابط برنامه‌نویسی
User interface	رابط کاربر
System administrator	راهبر سیستم
Online help	راهنمای برخط
Time event	رخداد زمانی

Mission	رسالت
Barcode reader	رمزینه‌خوان
Log	رویدادنگاری
Platform	سکو
Scenario	سناریو
Pseudo state	شبه وضعیت
Icon	شمايل
Include	شمول
Interface identification	شناسایی رابط
Object	شئ
Code page	صفحه کد
Design	طراحی
Signal	سیگنال
Send signal	سیگنال ارسال
Receive signal	سیگنال دریافت
Root cause	علت اصلی
Cause-and-Effect	علت و معلول
Action	عمل
Business process	فرایند کسب‌وکار
Activity	فعالیت
Technology	فناوری
Adaptability	قابلیت انعطاف
Testability	قابلیت آزمون
Reliability	قابلیت اطمینان
Supportability	قابلیت پشتیبانی
Configurability	قابلیت پیکربندی
Localizability	قابلیت محلی‌سازی
Installability	قابلیت نصب

Maintainability	قابلیت نگهداری
Business rule	قاعده کسب و کار
Naming convention	قرارداد نام‌گذاری
Constraint	قید
Performance	کارایی
Business function	کارکرد کسب و کار
Requirements workshop	کارگاه نیازمندی‌ها
Stereotype	کلیشه
Actor	کنشگر
Transition	گذار
Driver	گرداننده
Object flow	گردش شیء
Control flow	گردش کنترل
Focus Group	گروه متمرکز
Extend	گسترش
Pin	گیره
Product	محصول
Work product	محصول کاری
Requirements management	مدیریت نیازمندی‌ها
Problem	مسئله
Problem of problem	مسئله پشت مسئله
Product requirement document	مستند نیازمندی محصول
Observation	مشاهده
Supplementary specifications	مشخصات تکمیلی
Use case specification	مشخصات مورد کاربرد
Interview	مصطفی
Open interview	مصطفی باز
Closed interview	مصطفی بسته

Assumptions and dependencies	مفروضات و وابستگی‌ها
Audit	ممیزی
Expansion region	منطقه توسعه
Reverse engineering	مهندسی معکوس
Entity	موجودیت
Use case	مورد کاربرد
Primary use case	مورد کاربرد اصلی
Secondary use case	مورد کاربرد فرعی
Extension use case	مورد کاربرد گسترش یافته
Inclusion use case	مورد کاربرد مشمول
Central buffer	میانگیر مرکزی
MTTR- Mean Time To Repair	میانگین زمان تعمیر
MTBF- Mean Time Between Failures	میانگین زمانی بین خرابی‌ها
Asynchronous	ناهمگام
Token	نشانه
Discipline	نظام
Use case points	نقاط مورد کاربرد
Storyboarding	نقالی
Starting point	نقطه آغاز
Ending point	نقطه پایان
Exit point	نقطه خروج
Function-point	نقطه کارکرد
Entry point	نقطه ورود
Maintenance	نگهداری
Fishbone diagram	نمودار استخوان ماهی
Block diagram	نمودار بلوکی
Pareto chart	نمودار پارتو
Activity diagram	نمودار فعالیت

Class diagram	نمودار کلاس
Context diagram	نمودار متن
Use case diagram	نمودار مورد کاربرد
State diagram	نمودار وضعیت
Prototype	نمونه اولیه، پیش‌نمون
Prototyping	نمونه‌سازی
Need	نیاز
Requirement	نیازمندی
Design requirement	نیازمندی طراحی
Non-functional requirement	نیازمندی غیر کارکرده
Functional requirement	نیازمندی کارکرده
Software requirement	نیازمندی نرم‌افزاری
Synchronizer	همگام‌ساز
State	وضعیت
Initial state	وضعیت آغازین
Final state	وضعیت پایانی
Composite state	وضعیت مرکب
Feature	ویژگی

۲- واژه‌نامه انگلیسی به فارسی

"Yes, But" syndrome

آفت «بله، اما»

Action

عمل

Activity

فعالیت

Activity diagram

نمودار فعالیت

Actor

کنشگر

Adaptability

قابلیت انطباق

Analysis

تحلیل

Application programming interface (API)

رابط برنامه‌نویسی

Assumptions and dependencies

مفروضات و وابستگی‌ها

Asynchronous

ناهمگام

Audit

ممیزی

Authentication

احراز هویت

Availability

دسترس پذیری

Barcode reader

رمزینه‌خوان

Block diagram

نمودار بلوکی

Brainstorming

توفان ذهنی

Business function

کارکرد کسب و کار

Business process

فرایند کسب و کار

Business rule

قاعده کسب و کار

Cause-and-Effect

علت و معلول

Central buffer

میانگیر مرکزی

Choice

انتخاب

Class diagram

نمودار کلاس

Closed interview

مصطفی به بسته

Code page

صفحه کد

Composite state

وضعیت مركب

Configurability	قابلیت پیکربندی
Connector	بست
Constraint	قيد
Context diagram	نمودار متن
Control flow	گردش کنترل
Copyright	حق تألیف
Data store	انباره داده
Decision	تصمیم
Deep history	تاریخچه عمیق
Deployment	استقرار
Design	طراحی
Design requirement	نیازمندی طراحی
Development priority	اولویت توسعه
Discipline	نظام
Documents analysis	تحلیل مستندات
Domain expert	خبره حوزه مسأله
Driver	گرداننده
Ending point	نقطه پایان
Entity	موجودیت
Entry point	نقطه ورود
Exit point	نقطه خروج
Expansion region	منطقه توسعه
Extend	گسترش
Extendiblity	توسعه‌پذیری
Extension use case	مورد کاربرد گسترش یافته
Feature	ویژگی
Final state	وضعیت پایانی
Fishbone diagram	نمودار استخوان ماهی

Flow final	پایان گردش
Focus Group	گروه متمرکز
Fork	انشعاب
Functional requirement	نیازمندی کارکرده
Function-point	نقشه کارکرد
Generalization	تعمیم
Icon	شمايل
Include	شمول
Inclusion use case	مورددکاربرد مشمول
Initial state	وضعیت آغازین
Installability	قابلیت نصب
Interface	رابط
Interface identification	شناسایی رابط
Interview	مصطفحه
Join	الحق
Junction	پیوند
Lifecycle	چرخه زندگی
Localizability	قابلیت محلی سازی
Log	رویدادنگاری
Maintainability	قابلیت نگهداری
Maintenance	نگهداری
Merge	ادغام
Mission	رسالت
MTBF- Mean Time Between Failures	میانگین زمانی بین خرابی‌ها
MTTR- Mean Time To Repair	میانگین زمان تعمیر
Naming convention	قرارداد نام‌گذاری
Need	نیاز
Non-functional requiremnt	نیازمندی غیرکارکرده

Object	شیء
Object flow	گردش شیء
Observation	مشاهده
Online help	راهنمای برخط
Open interview	مصاحبه باز
Pareto chart	نمودار پارتو
Partition	جداساز
Partition	جداساز
Performance	کارایی
Pin	گیره
Platform	سکو
Policy	خط مشی
Precedence and priority	ترتیب و اولویت‌بندی
Primary use case	مورد کاربرد اصلی
Problem	مسئله
Problem analysis	تحلیل مسئله
Problem of problem	مسئله پشت مسئله
Product	محصول
Product positioning	جاگاه محصول
Product requirement document	مستند نیازمندی محصول
Prototype	نمونه اولیه، پیش‌نمون
Prototyping	نمونه‌سازی
Pseudo state	شبه وضعیت
Quality attributes	خصوصیات کیفی
Quality goals	اهداف کیفی
Receive signal	سیگنال دریافت
Reliability	قابلیت اطمینان
Repairability	تعمیرپذیری

Requirement	نیازمندی
Requirements elicitation	استخراج نیازمندی‌ها
Requirements management	مدیریت نیازمندی‌ها
Requirements workshop	کارگاه نیازمندی‌ها
Reverse engineering	مهندسی معکوس
RFP- Request for proposal	درخواست برای پیشنهاد
Role playing	ایفای نقش
Root cuase	علت اصلی
Scanner	پویش‌گر
Scenario	سناریو
Secondary use case	مورد کاربرد فرعی
Send signal	سیگنال ارسال
Shallow history	تاریخچه سطحی
Signal	سیگنال
Software requirement	نیازمندی نرم‌افزاری
Stakeholder	ذینفع
Starting point	نقطه آغاز
State	وضعیت
State diagram	نمودار وضعیت
Stereotype	کلیشه
Storyboarding	نقالی
Subject matter expert	خبره موضوع
Suplementary specifications	مشخصات تکمیلی
Supportability	قابلیت پشتیبانی
Survey/Questionarie	تحقیق/پرسشنامه
Synchronizer	همگام‌ساز
System administrator	راهبر سیستم
System attributes	خصوصیات سیستم

	فناوری
Technology	
Terminate	پایان‌گر
Test	آزمون
Test case	آزمایه
Testability	قابلیت آزمون
The "User and the Developer" syndrome	آفت کاربر - توسعه‌دهنده
Throughput	توان عملیاتی
Time event	رخداد زمانی
Token	نشانه
Transition	گذار
Usability	خوش‌کاری
Use case	مورد کاربرد
Use case diagram	نمودار مورد کاربرد
Use case points	نقاط مورد کاربرد
Use case specification	مشخصات مورد کاربرد
User interface	رابط کاربر
Vision	چشم‌انداز
Work product	محصول کاری

Part 5: Complementary Topics

In this part, two complementary topics are discussed which include important challenges in, and techniques of requirements elicitation.

Chapter 9: The Challenge of Requirements Elicitation

In this chapter, challenges in eliciting requirements as well as solutions for handling these challenges are explained.

Chapter 10: Requirements Elicitation Techniques

This chapter discusses some of the techniques for requirements elicitation. We also provide a high-level overview of a tool-box for these techniques.

Part 6: UML Diagrams

In this part, the Activity and the State diagrams are explained. These diagrams are UML diagrams which are frequently used during the analysis of requirements. This part contains two chapters.

Chapter 11: The Activity Diagram

Chapter 12: The State Diagram

Book homepage: www.somamos-co.ir

Chapter 3: Problem analysis method

In this chapter, the term “Problem” is first defined and, afterwards, the importance of identifying problems, root cause analysis, and the steps of problem analysis are explained.

Part 3: Understand Stakeholders' Needs

The focus of this part is on identifying the key stakeholders’ needs through gathering information about their desired software. In addition, how to elicit and document the business rules are also discussed.

This part consists of two chapters;

Chapter 4: Methods to identify stakeholders’ needs

The importance of identifying the stakeholder’s needs and the approach to outlining a Vision document are explained through this chapter.

Chapter 5: Maintaining Business Rules

Owing to the necessity of covering some business rules by software systems, the definition, classification and the method of identifying and documenting these rules are discussed in this chapter.

Part 4: Defining the System

The purpose of this part is to define the detailed requirements of software. This includes specifying use cases along with Supplementary Specification document. This part contains three chapters:

Chapter 6: Use-case model

Throughout this book, use cases are utilized as a technique for eliciting and documenting requirements. This chapter defines and explains use case model.

Chapter 7: Developing use-case model

This chapter explains a method to develop use-case model and its related work products.

Chapter 8: Developing Supplementary Specifications

This chapter outlines a method that specifies requirements which cannot (are not possible to) be specified as use cases.

Introduction

Generally, there are many challenges in software system development processes in small-, medium-, and large-scale Iranian companies. Two well-known challenges are: The complexity of software development methodologies is carried out regardless of the principles of Software Engineering. These challenges are usually due to the lack of a simple, but deep, perception of the Software Engineering concepts. Most of the time, only adhering to these simple concepts can lead/conduct a project to a successful end, without having to face the complexity of software development methodologies.

Another cause of such challenges is the complexity and the difficulty of mapping and performing activities belonging to a development process, such as RUP. Such complexity and difficulty push development teams towards a product-driven attitude (way of thinking) rather than a task-driven one. For instance, development team focuses on the preparation of a “Vision document” instead of considering a set of tasks and activities that lead to its production. This book introduces a role-based framework and a simple model for eliciting and analyzing requirements.

Parts and chapters of the book:

This book is presented in six parts. The parts have been ordered so that the correlation and the cohesiveness of topics are preserved. Complementary topics, have been brought out (put away) as final parts; preceded by topics on how to conduct presented processes.

The parts and chapters are as follows:

Part 1: Introduction

This part provides an introduction to various topics on requirements domain and requirements management scope. This part contains two chapters;

Chapter 1: The importance of Requirements

Through this chapter, the importance of Requirements discipline and its effect on the success of software projects are discussed.

Chapter 2: Basic Definitions

In this chapter, the concepts, definitions, and keywords of Requirement discipline are explained.

Part 2: Analyze the Problem

This part describes the first step in the analysis of requirements, i.e. Problem Analysis, and contains one chapter;

Software Requirements Analysis

A Practical Approach

By:

Yusef Mehrdad Bibalan

Pooya Shahbazian

Mozafar Iraf

2013

ISBN: 978-964-388-400-0



لهم اذن بر این کتاب و آن را در میان اهالی
کردستان بسیاری از این کتابها که در این کتابخانه
در این سالهای اخیر از این کتابخانه خریداری شده
باشند.
آنها را که از این کتابخانه خریداری نکردند
آنها را که از این کتابخانه خریداری نکردند
آنها را که از این کتابخانه خریداری نکردند
آنها را که از این کتابخانه خریداری نکردند

آنها را که از این کتابخانه خریداری نکردند
آنها را که از این کتابخانه خریداری نکردند
آنها را که از این کتابخانه خریداری نکردند
آنها را که از این کتابخانه خریداری نکردند

انتشارات صفار

از استادان محترم دانشگاه‌هایی که در رشته‌های
کامپیوتر، فنی مهندسی، علوم پایه و مدیریت تدریس
من کنند جهت تألیف و ترجمه دعوت به همکاری
من کنند.

همچین به استادان محترمی که کتاب‌های
دانشگاهی این انتشارات را به عنوان کتاب‌های
درسی جهت تدریس برگزینند یک جلد کتاب به صورت
رایگان اهدا من شود.

جهت اطلاع بیشتر با شماره تلفن‌های ۶۶۴۰۸۴۸۷
- ۶۶۴۱۳۴۳۶ و جهت مشاوره چاپ با شماره همراه
۰۹۱۲-۱۰۷۳۰۰۳ تماس حاصل فرمایید.

Software Requirements Analysis

A Practical Approach

اگر نیازمندی‌ها را به درستی شناسایی نکنید
خوب انجام دادن بقیه پروژه، دیگر اهمیتی نخواهد داشت.

این کتاب به شما می‌آموزد که تحلیل نیازمندی‌ها را در پروژه‌های نرم افزاری به چه روشی انجام دهید.

کتاب، حاصل اجرای چند باره روش در پروژه‌های نرم افزاری و تجربیات آموزشی در دانشگاه‌ها شرکت‌های نرم افزاری و کارگاه‌های تخصصی است.

در این کتاب آمده است:

- ادبیات و اهمیت حوزه نیازمندی‌ها
- چگونگی شناسایی نیازهای مشتریان و کاربران
- نحوه شناسایی و تدوین قواعد کسب و کار
- روش شناسایی و تدوین ویژگی‌های موارد کاربرد سیستم
- آفت‌ها و تکنیک‌های استخراج نیازمندی‌ها
- تشریح کاربرد UML در تحلیل نیازمندی‌ها به همراه مثال
- قالب‌های مستندات
- ارائه نمونه‌هایی از یک سیستم در کل کتاب

برای خرید
به آدرس زیر مراجعه کنید
www.saffarpublishing.ir



ISBN 978-964-388-400-0

