

# WEB APP DESIGN

درس مهندسی نرم افزار ۲  
دکتر احمد عبدالله زاده بارفروش  
تهیه کننده : ملیحه هاشمی

# ویژگی های متمایز WEB APP

❖ تفاوت web app با نرم افزارهای conventional از ابعاد زیر قابل بررسی است:

➤ Application realated characteristics

➤ Usage related characteristics

➤ Development related characteristics

➤ Evolution related characteristics

# ویژگی های متمایز WEB APP

Content is king

Application based characteristics ❖

➤ محتوا (content):

- سیستم های تحت وب صرف نظر از کارکردهایی که فراهم می آورند، بسیار وابسته به محتوا هستند.
- شامل داده های ساختاریافته ی پایگاه داده + داده های ساختارنیافته یا نیمه ساختاریافته مانند توصیفات متنی یا اطلاعات چند رسانه ای هستند.
- در سیستم های تحت وب محتوا بسیار پویا است و به طور مداوم به روز رسانی می باشد.
- در سیستم های تحت وب محتوا باید از کیفیت بالایی برخوردار باشد، مانند: دقت، سازگاری، قابل اعتماد بودن، به روز بودن

➤ ساختاردهی محتوا از طریق hypertext

➤ زیبایی

➤ امنیت

# ویژگی های متمایز WEB APP

## Usage related characteristics ❖

### :Unpredictable load ➤

- تعداد کاربران برنامه‌ی کاربردی تحت وب ممکن است در هر زمان متفاوت باشد. برای مثال ممکن است در یک روز ۵۰ درخواست در طول ساعت فرستاده شود، اما در روز بعد ۱۵۰ درخواست در طول ساعت ارسال گردد.

### :Unpredictable user ➤

- برنامه‌های کاربردی تحت وب، مورد استفاده‌ی طیف وسیع و متفاوتی از کاربران قرار می‌گیرند که هر یک دارای نیازمندی‌ها، انتظارات و توانایی‌های گوناگون و همچنین الگوهای رفتاری ناشناخته هستند. بنابراین واسط کاربری و تعاملات با سیستم باید طوری طراحی شوند که نیازمندی‌های (خصوصاً نیازمندی‌های وابسته به useability) این کاربران گسترده و ناشناخته را برآورده کند.
- زیرساخت‌های تکنیکی غیرقابل پیش‌بینی ابزارهای در دسترس کاربران پایانی در توانایی‌های سخت‌افزاری و نرم‌افزاری مانند سایز نمایش، توان محاسباتی و نسخه‌ی مرورگر با هم متفاوتند و این یکی دیگر از مسائلی که تولید این گونه سیستم‌ها را دچار مشکل می‌نماید.

# ویژگی های متمایز WEB APP

## Development related characteristics ❖

### تیم تولید: ➤

■ جهت تولید سیستم های تحت وب، تیمی متشکل از افرادی با طیف گسترده ای از مهارت ها و تخصص ها در حوزه های مختلف نیاز است. تخصص هایی در زمینه مهندسی نرم افزار، مهندسی اطلاعات، طراحی گرافیکی، مدیریت شبکه و ... در نتیجه این تیم نسبت به تیم مورد نیاز برای برنامه های نرم افزاری غیر وب افراد و نقش های بیشتری را شامل می گردد و مدیریت و هماهنگی آن نیازمند توجه به این مسئله است.

### محیط تولید: ➤

■ زیرساخت تکنیکی مورد استفاده برای تولید یک سیستم تحت وب با درجه ی بالایی از نوسانات و ناهمگونی ها مشخص می شود. به عبارتی تولید سیستم تحت وب، متکی بر طیف گسترده ای از مولفه ها مانند وب سرور، سرور برنامه های کاربردی، مولفه های کتابخانه ای، سیستم های پایگاه داده، چارچوب- های منتشر شده، زبان ها و استانداردهای برنامه نویسی می باشد. با توجه به افزایش فشار زمان عرضه به بازار، این مولفه ها اغلب ناآزموده بوده و پایداری، قابلیت اطمینان و کارکردهای موردنظر را فراهم نمی آورند. ( مسئله ناپایداری تکنولوژی در محیط تولید)

# ویژگی های متمایز WEB APP

## Evolution related characteristics ❖

- محتوا و کارکردهای برنامه‌های کاربردی تحت وب به طور مکرر بدون نسخه‌های مشخص و با دوره‌های نگهداری روزانه یا حتی ساعتی، به هنگام می‌شوند.
- بنابراین برخلاف نرم افزارهای conventional که از طریق تعدادی release برنامه‌ریزی شده و مشخص تکامل می‌یابند، برنامه‌های کاربردی تحت وب به طور مداوم تغییر می‌یابند.

# اهداف طراحی وب

## ❖ سادگی

- در طراحی سایت نباید محتوای زیادی در داخل صفحات گنجانده، انیمیشن‌های زیادی استفاده نمود و یا به میزان زیادی تصاویر و جلوه‌های بصری به کار گرفت. (باید به سادگی و تعادل عناصر به کار رفته توجه کرد.)
- محتوای سایت باید حاوی اطلاعات مفید و مختصر باشد. همچنین نحوه‌ی ارائه‌ی اطلاعات به کاربر (از طریق ویدیو، متن، اشکال گرافیکی و ...) باید با نوع این اطلاعات تناسب داشته باشد.
- نحوه‌ی پیمایش در سایت باید برای کاربر به راحتی قابل فهم باشد.
- کارکردهای ارائه شده باید به راحتی برای کاربر قابل استفاده باشند.
- باید توجه کرد که پیمایش سایت‌های ساده‌تر برای کاربر راحت‌تر است. همچنین سایت‌های ساده‌تر سریع‌تر load می‌شوند.

## ❖ Identity

- عناصر به کار رفته جهت طراحی زیبایی سایت، طراحی واسط کاربری، طراحی پیمایشی برنامه کاربردی تحت وب باید با کاربرد مورد نظر تطابق داشته باشد. (برای مثال طراحی یک سایت موزیک با یک سایت خدمات پزشکی متفاوت است.)
- بنابراین طراحی انجام شده برای یک برنامه کاربردی تحت وب باید به صورتی باشد که به آن یک هویت مشخص مطابق با دامنه کاربرد مورد نظر بدهد.

7

# اهداف طراحی وب

## ❖ سازگاری

- سازگاری در محتوا باید مورد توجه قرار گیرد. (قالب‌بندی متن و style‌های به کارگرفته شده باید در محتوای سایت یکسان باشند؛ اشکال به کار رفته دارای ظاهر یکسان باشند، رنگ‌های به کار گرفته شده در رابطه با عناصر باید با یکدیگر سازگاری داشته باشند).
- طراحی معماری باید به نحوی انجام شده باشد که ساختار برنامه‌کاربردی در بخش‌های گوناگون آن سازگار باشد.
- طراحی واسط کاربری باید به نحوی صورت پذیرد که شیوه‌ی تعامل کاربر با برنامه کاربردی در موقعیت‌های مشابه یکسان باشد.
- مکانیسم‌های پیمایشی مورد استفاده باید در تمام طول برنامه کاربردی به صورت یکسان و سازگار مورد استفاده قرار گیرند.

## ❖ Robustness

- محتوا و کارکردهایی که در اختیار کاربر قرار می‌گیرد باید دقیق و بدون نقص باشد.
- به این ترتیب برنامه‌ی کاربردی تحت وب باید طوری طراحی شده باشد که بتواند خطاهای موجود در داده‌های ورودی، تعاملات غیرپیش‌بینی شده از طرف کاربر را اداره کند و در صورت بروز چنین مواردی محتوا و کارکردی که با نیازمندی‌های کاربر مطابقند را ارائه نمایند.

8



# اهداف طراحی وب

## Navigability ❖

- همانطور که گفته شد پیمایش در برنامه کاربردی تحت وب باید ساده و سازگار باشد. (باید به طریقی طراحی شود که برای کاربر قابل فهم بوده و قابل پیش‌بینی باشد).
- به عبارتی کاربران بدون اینکه به دنبال لینک‌های پیمایش و یا دستورالعملی برای این منظور باشند باید به راحتی نحوه حرکت و پیمایش در برنامه‌ی کاربردی را متوجه شوند.
- لینک‌های پیمایش باید در محل‌های قابل دسترسی، قابل پیش‌بینی قرار گیرند. (کاربر باید بتواند به راحتی لینک موردنظر خود را از میان لینک‌های موجود شناسایی نماید).
- در صورتی که از scroll در طراحی صفحه استفاده شده است، لینک‌ها باید در بالا و یا پایین صفحه قرار گیرند. (نه در بین محتوا، تا کاربر به راحتی آن‌ها را پیدا کند).

## Visual appeal ❖

- در بین تمام دسته‌بندی‌های نرم‌افزار، برنامه‌های کاربردی تحت وب به بیشترین میزان به جنبه‌های ظاهری و بصری وابسته هستند. بنابراین در این رابطه باید به زیبایی look&feel، صفحه‌بندی واسط کاربری، هماهنگی رنگ‌ها، تناسب متن، اشکال گرافیکی و مکانیسم‌های پیمایش توجه نمود).

# اهداف طراحی وب

## Compatibility ❖

- برنامه‌های کاربردی تحت وب در محیط‌های گوناگون (سخت افزار متفاوت، مرورگرهای متفاوت، سیستم عامل متفاوت و انواع گوناگون روش‌های اتصال به اینترنت با سرعت متفاوت) مورد استفاده قرار می‌گیرند.
- بنابراین در صورتیکه یک برنامه‌ی کاربردی در محیط‌های گوناگون به کار گرفته شود، باید طوری طراحی شود که نسبت به موارد بالا قابلیت سازگاری داشته باشد.

# کیفیت برنامه‌های کاربردی تحت وب



❖ یک دسته‌بندی مناسب برای بیان نیازمندی‌های برنامه‌های کاربردی تحت وب، دسته‌بندی **FURPS+** است.

- Functionality (کارکرد)
- Reliability (قابلیت اطمینان)
- Usability (قابلیت استفاده)
- Performance (کارایی)
- Supportability (قابلیت پشتیبانی)

➤ “+” در **FURPS+** شامل نیازمندی‌های:

- Design Constraints (محدودیت‌های طراحی)
- Implementation Requirements (نیازمندی‌های پیاده سازی)
- Interface Requirements (نیازمندی‌های واسط)
- Physical Requirements (نیازمندی‌های فیزیکی)

11

# کیفیت برنامه‌های کاربردی تحت وب

❖ برای مثال در رابطه با هر یک از ابعاد کیفیت برنامه کاربردی تحت وب می‌توان به موارد زیر توجه نمود.

کیفیت برنامه کاربردی تحت وب	Functionality	Searching and Retrieving Issues	Reliability	Correct link processing Dangling Links Invalid Links Unimplemented Links
		Navigation and browsing features		Error recovery
		Application domain-related features		User input validation and recovery
	Usability	Global Site Understandability Global Organization Scheme Site Map Table of Content Alphabetical Index Quality of Labeling System	Maintainability	Ease of correction
				Adaptability
				Extensibility
		On-line Feedback and Help Features Quality of Help Features Web-site Last Update Indicator Addresses Directory FAQ Feature On-line Feedback	Efficiency	Response time performance
				Page generation speed
		Interface and Aesthetic Features		Graphics generation speed
		Special feature Foreign Language Support		

## کیفیت برنامه‌های کاربردی تحت وب

❖ همچنین در رابطه با برنامه‌های کاربردی تحت وب، توجه به صفات کیفی زیر از اهمیت بالایی برخوردار است:

➤ امنیت: با توجه به اینکه بسیاری از برنامه‌های کاربردی تحت وب، به پایگاه داده‌های حساس مرتبط می‌شوند و یا اطلاعات مهم کاربران دریافت و نگه داری می‌کنند، امنیت برنامه‌های کاربردی تحت وب از اهمیت بسیار بالایی برخوردار است. در این رابطه برنامه‌های کاربردی تحت وب باید دسترسی کاربران به داده‌ها و کارکردها را کنترل کرده و مانع از خرابی داده‌ها و عدم ارائه‌ی سرویس به دلیل دسترسی‌های غیرمجاز گردد. (این صفت کیفی در دسته بندی FURPS+ در بعد functionality قرار می‌گیرد).

➤ دسترسی‌پذیری (availability): در صورتی که برنامه‌ی کاربردی تحت وب تمام محتوا و کارکردهای موردنظر کاربران را پیاده‌سازی کرده باشد، اما در مواقع نیاز در دسترس نباشد، در عمل برای کاربران قابل استفاده نیست. دسترسی‌پذیری بیان کننده‌ی احتمال این است که سیستم در مواقع نیاز در دسترس کاربر باشد. (این صفت کیفی در دسته بندی FURPS+ در بعد reliability قرار می‌گیرد).

## کیفیت برنامه‌های کاربردی تحت وب

❖ همچنین در رابطه با برنامه‌های کاربردی تحت وب، توجه به صفات کیفی زیر از اهمیت بالایی برخوردار است:

- **مقیاس‌پذیری (scalability):** با توجه به اینکه یکی از ویژگی‌های برنامه‌های کاربردی تحت میزان load غیرقابل پیشبینی است، مقیاس‌پذیری یکی از ویژگی‌های اصلی این گونه سیستم‌ها می‌باشد. یک سیستم مقیاس‌پذیر طوری طراحی می‌شود که می‌تواند در صورت نیاز ۱۰۰، ۱۰۰۰، ۱۰۰۰۰ و یا بیشتر کاربر را پشتیبانی نماید. (این صفت کیفی در دسته بندی FURPS+ در بعد performance قرار می‌گیرد).
- **زمان عرضه به بازار:** اگرچه زمان عرضه به بازار، یک صفت کیفی از بعد تکنیکی نیست، اما با توجه به فشار رقابت یکی از مواردی است که از بعد کسب‌وکار دارای اهمیت است.

# کیفیت برنامه‌های کاربردی تحت وب

## ❖ کیفیت محتوا:

- با توجه به اینکه یکی از ویژگی‌های متمایز کننده برنامه‌های کاربردی تحت وب، اهمیت محتوا در این گونه سیستم‌هاست، بنابراین کیفیت محتوا در این گونه سیستم‌ها تعیین کننده است.
- از جمله ویژگی‌های تاثیرگذار در کیفیت محتوای به روز بودن (timeliness)، دقت (accuracy)، کامل بودن (completeness)، صداقت (veracity) می‌باشد.
- همچنین با توجه به سوالات زیر نیز می‌توان کیفیت برنامه‌های کاربردی تحت وب را سنجید:
  - آیا محتوا و نحوه‌ی ارائه آن به کاربر از پایداری برخوردار است؟ (آیا محتوا از طریق url تعیین شده همواره قابل دسترسی است؟)
  - آیا مشخص است که محتوا آخرین بار در چه تاریخی تغییر کرده است؟ آیا مشخص است کدام بخش از محتوا تغییر کرده است؟
  - آیا محتوا در ساختار مناسب سازمان دهی شده است؟ آیا به خوبی ایندکس شده است؟ به راحتی قابل دسترسی است؟
  - آیا محتوای ارائه شده منحصر به فرد است؟ به عبارتی آیا برنامه کاربردی تحت وب به وسیله محتوا، اطلاعات منحصر به فردی به کاربر ارائه می‌دهد؟
  - آیا می‌توان به راحتی عمق و محدوده محتوا را تعیین نمود و از این طریق از برآورده‌سازی نیاز کاربر اطمینان حاصل کرد؟

15

## هرم طراحی برنامه‌های کاربردی تحت وب



16



# طراحی واسط کاربری

## ❖ واسط کاربری باید به نحوی طراحی شود که به سوالات زیر پاسخ دهد:

➤ **Where am I?** در صورتی که از یک صفحه bookmark شده استفاده شود، باید طراحی واسط کاربری به نحوی صورت پذیرفته باشد که context این صفحه مشخص باشد. (لینکی به صفحه اصلی سایت وجود داشته باشد و کاربر بتواند از موقعیت فعلی خود در سلسله مراتب محتوا آگاه شود).

■ به عبارتی در صورتی که طراحی برنامه کاربردی به نحوی انجام شده است که کاربران از طریق entry poin های مختلف می توانند به آن دسترسی داشته باشند، باید مطمئن باشیم که طراحی هر صفحه به صورتی است که کاربران بتوانند صفحات دیگر را پیمایش کنند.

➤ **What can I do now?** واسط کاربری همواره باید به کاربر در تشخیص گزینه های عملیاتی کمک کند: این که کاربر در حال حاضر از چه کارکردهایی می تواند استفاده نماید، چه لینک های توسط او قابل استفاده هستند و چه بخش هایی از محتوای برنامه ی کاربردی می توانند به او مرتبط باشند.

➤ **Where have I been, where am I going?** واسط کاربری باید طوری طراحی شود که به پیمایش کاربر در برنامه کاربردی کمک کند. از این رو، واسط کاربری باید یک نقشه ای از سایت فراهم کند که مشخص می کند که کاربر در حال حاضر در کجا قرار دارد و برای این که بخش های دیگر برود، چه مسیرهایی را می تواند انتخاب کند.

✓ با پاسخ گویی به این سوالات، واسط کاربری امکان استفاده از کارکردها و محتوای فراهم شده توسط برنامه کاربردی را فراهم می آورد، کاربر را در انجام تعاملات با برنامه کاربردی راهنمایی می کند و گزینه های پیمایشی و محتوا را به نحوی سازمان دهی شده در اختیار کاربر قرار می دهد.

17

# اصول طراحی واسط کاربری

## ❖ پیش‌بینی (Anticipation):

- برنامه کاربردی تحت وب باید طوری طراحی شود که بتواند حرکت بعدی کاربر را پیش‌بینی نماید. به این ترتیب اطلاعات و کارکردها در هر گامی که مورد نیاز باشد، در دسترس او قرار می‌گیرد. برای مثال:
- در سیستم پورتال آموزشی زمانی که کاربر دروس مورد نظر را انتخاب کرد، ممکن است بخواهد برنامه هفتگی خود را مشاهده نماید. بنابراین طراح واسط کاربری می‌تواند با پیش‌بینی این مورد، در صفحه دروس انتخاب شده، امکان دسترسی به برنامه هفتگی را به نحوی فراهم کند.
- در سیستم فروش وسایل دیجیتال، وقتی که کاربر یک محصول مانند پرینتر، اسکنر و ... را خریداری کرد احتمالاً مایل است درایورهای آن نیز پیدا کند. بنابراین طراح می‌تواند با پیش‌بینی این امر پس از تکمیل خرید امکان پیمایش به صفحه دانلود درایور را برای کاربر فراهم کند.

## ❖ Communication:

- واسط کاربری باید وضعیت هر فعالیتی که به کاربر مرتبط است را به وی اطلاع دهد.
- این وضعیت باید دقیق و به روز باشد.
- همچنین این اطلاعات باید به راحتی در اختیار کاربر قرار گیرند و کاربر مجبور نباشد برای مشاهده این اطلاعات وضعیتی، فعالیت خود را متوقف کرده و پس از آن به این اطلاعات دسترسی داشته باشند. ( این اطلاعات باید در پس زمینه قابل مشاهده باشند، مثلاً در گوشه پایین صفحه)

18

# اصول طراحی واسط کاربری

## ❖ سازگاری (Consistency):

- استفاده از روش‌های پیمایش، منوها، آیکن‌ها، شکل‌ها و رنگ‌ها باید در تمام بخش‌های برنامه کاربردی تحت وب، سازگار باشد. ( برای مثال در صورتی که لینک‌ها با رنگ آبی و underline مشخص می‌شوند، برای سایر کلماتی که به لینکی اشاره ندارند از این style استفاده نشود.)
- واسط کاربری باید با انتظارات کاربر سازگار باشد. علائم و آیکن‌های به کار رفته در واسط کاربری باید مطابق با کاربردهای معمول باشد. (به عبارتی آیکن‌هایی که برای کاربر آشنا هستند برای منظور مشابه استفاده شوند.)
- در واسط کاربری، کارکردهای متفاوت باید به شکل متفاوت نمایش داده شوند. (مثلا، آیکن‌های متفاوتی داشته باشند.)

## ❖ Controlled autonomy:

- برنامه کاربردی تحت وب باید طوری طراحی شده باشد که امکان حرکت کاربر در بخش‌های مختلف برنامه‌های کاربردی و استفاده از کارکردهای مختلف را با توجه به اهداف وی سهولت بخشند.
- کاربر باید بتواند در نحوه بهره‌گیری از برنامه کاربردی اختیار داشته باشد. (در انتخاب ظاهر برنامه، ترتیب به کارگیری کارکردها، پیمایش بخش‌های مختلف و ...)
- با این وجود نحوه ی استفاده کاربر از برنامه کاربردی باید به طریقی کنترل شود که قوانین تعیین شده نقض نشوند. (برای مثال مطمئن شویم که کاربر به بخش‌هایی که مجاز به استفاده از آن نیست دسترسی نداشته باشد، بنابراین کاربر نباید با استفاده از هیچ مکانیزم پیمایشی به این بخش‌ها منتقل شود.)

19

# اصول طراحی واسط کاربری

## ❖ کارایی (efficiency):

- طراحی واسط کاربری باید به نحوی انجام شود که کارایی عملکرد کاربران افزایش یابد نه کارایی برنامه کامپیوتر.
  - برای مثال در طراحی یک فرم در سیستم آموزش، انتخاب دانشکده از میان لیست دانشکده‌ها برای کاربر آسان‌تر تا این که نام دانشکده را وارد کند. بنابراین استفاده از منوهای کشویی نسبت به یک فیلد متنی مناسب‌تر است. در صورتی که از دید برنامه کامپیوتری به کارگیری فیلد متنی نسبت به منوهای کشویی احتیاج به خط کد کمتری دارد.
- برای افزایش کارایی عملکرد کاربران، پیغام‌های خطا باید طوری نوشته شده باشند که نشان دهند چه چیزی موجب بروز خطا شده است و چگونه می‌توان این خطا را برطرف نمود.

## ❖ Human interface object

- برای واسط کاربری برنامه‌های کاربردی تحت وب، تعداد زیادی interface object ارائه شده است. (مانند menu ها، button ها، folder ها، stadrad icon ها و ...)
- روش های استاندارد مشخصی برای استفاده از این عناصر برنامه نویسی وجود دارد، بنابراین در واسط کاربری برنامه‌های کاربردی تحت وب باید از این روش های استاندارد استفاده شود. (برای مثال یک button باید فشرده شود، باید بتوان برای انتخاب مقادیر موردنظر روی یک slider، drag کرد.)
- استفاده از عناصر برنامه نویسی منجر به رفتار و نتایج استاندارد شود. ( برای مثال با فشردن یک button ، یک فرم submit می‌شود.)

20

# اصول طراحی واسط کاربری

## ❖ Learnability

- واسط کاربری برنامه کاربردی تحت وب باید طوری طراحی شود که زمان یادگیری کاربر برای کار با آن کم باشد.
- همچنین واسط کاربری باید طوری طراحی شود که اگر کاربر کار با آن را یاد گرفت، در صورت ایجاد تغییر در آن، نیاز به یادگیری مجدد کاربر به حداقل برسد.

## ❖ Latency reduction

- برنامه کاربردی تحت وب باید طوری طراحی شده باشد که در هنگام اجرای یک عملیات زمان بر و پیچیده کاربر را منتظر نگه ندارد، بلکه به او امکان بدهد که در این حین تا تکمیل عملیات موردنظر، بتواند کارهای دیگری انجام دهد. (قرار دادن عملیات پیچیده در پس زمینه و مخفی کردن تاخیر از دید کاربر)
- با توجه به پایین بودن سرعت اینترنت، در صورتی که کاربر پس از فشردن یک دکمه نتیجه‌ی کارکرد موردنظر خود را دریافت نکند، ممکن است به طور مجدد و چندین بار این دکمه را فشار دهد. این امر موجب ارسال چندین باره‌ی درخواست کاربر و پایین آمدن بیشتر سرعت می‌شود. بنابراین واسط کاربری برنامه کاربردی تحت وب باید طوری طراحی شود که ارسال چندین باره‌ی درخواست بر اثر کلیک‌های مداوم کاربر را کنترل کند.

# اصول طراحی واسط کاربری

## ❖ Latency reduction (ادامه) :

➤ علاوه بر سعی در کاهش تاخیر، کاربر باید به نوعی از میزان تاخیر آگاه شود تا بداند که در برنامه چه اتفاقی می افتد. ( برای مثال استفاده از آیکن انتظار یا یک process bar که نشان میدهد زمانی که این نوار تکمیل شود، عملیات انجام خواهد شد.)



➤ همچنین در صورتی که کاربر باید بیشتر از چند ثانیه منتظر بماند، زمان انتظار که مرتباً در حال کاهش باید به او نشان داده شود.

## ❖ قانون Fitt:

➤ از قوانینی که در طراحی واسط کاربری مورد توجه می باشد، قانون Fitt است. این قانون به این صورت بیان می شود: «زمان مورد نیاز برای دسترسی به یک عنصر هدف (target object) در صفحه، تابعی از اندازه و فاصله اشاره گر تا آن عنصر

ثابت های  $a, b$  به طور تجربی به دست می آیند و وابسته به device ای هستند که کاربر با آن در تعامل است.

$$Time (Msecond) = a + b \log_2(D/s + 1)$$



# AESTHETIC DESIGN

❖ طراحی گرافیکی برنامه کاربردی تحت وب و جنبه‌های مربوط به look & feel آن را شامل می‌شود.

❖ هدف آن افزایش زیبایی ظاهری یک برنامه کاربردی وب است که سایر جنبه‌های تکنیکی آن را تکمیل می‌کند.

❖ بدون توجه به این بعد از طراحی هر چند که یک برنامه کاربردی از لحاظ کارکردی و محتوایی برای کاربران قابل استفاده است اما ممکن است مورد توجه آن‌ها قرار نگیرد.

بسیاری از کاربران به سرعت یک سایت را از روی ظاهر آن ارزیابی می‌کنند.

❖ توجه به نکات زیر در صفحه‌بندی، می‌تواند در زیبایی یک برنامه کاربردی وب موثر باشد:

➤ استفاده از فضاهای خالی: توصیه نمی‌شود که تمام بخش‌های یک صفحه وب را از اطلاعات پر کرد. زیرا کاربر اطلاعات و کارکردهای موردنظر خود را به سختی پیدا می‌کند و سازمان‌دهی و نظم صفحه به هم می‌خورد.

➤ تاکید بر محتوا: با توجه به این که هدف اصلی کاربر از مراجعه به یک صفحه وب، استفاده از محتوا و کارکردهای ارائه شده در آن می‌باشد، ۸۰٪ فضای صفحه وب باید به محتوا اختصاص داشته باشد، و ما بقی فضا به منوها، Navigation و سایر ویژگی‌ها اختصاص داشته باشد.

➤ عناصر موجود در صفحه باید از گوشه‌ی سمت چپ بالا (در زبان‌های راست به چپ از گوشه‌ی سمت راست) شروع و در گوشه‌ی سمت راست پایین (در زبان‌های راست به چپ در گوشه‌ی سمت چپ) به پایان برسند: زیرا بسیاری از کاربران علاقه مند هستند که صفحات وب را مانند صفحات کتاب پیمایش کنند.

# AESTHETIC DESIGN

## ❖ ادامه نکات مربوط به صفحه‌بندی:

- **گروه بندی فضای صفحه:** فضای صفحه وب به درستی تقسیم‌بندی شده باشد به طوری که بخش‌های محتوا، پیمایش و کارکردهای سایت هر یک در گروه مجزایی در کنار هم قرار داشته باشند.
- **عدم افزایش فضای صفحه با استفاده از scroll bar:** اگرچه استفاده از scroll bar در برخی موارد ضروری به نظر می‌رسد اما مطالعات نشان داده است که کاربران ترجیح می‌دهند صفحه دارای scroll نباشد. بنابراین بهتر است به جای استفاده از scroll محتوا را کاهش داد، یا آن را در چندین صفحه به کاربر ارائه داد.
- **توجه به رزولوشن و اندازه‌ی صفحه مرورگر در هنگام صفحه بندی:** اندازه‌های تعیین شده برای بخش‌های مختلف سایت نباید fixed باشند بلکه باید به صورت نسبی و براساس درصد بیان شوند.
- **اگر در طراحی صفحه از تصاویر استفاده شده است، آن‌ها را در اندازه‌ی کوچک قرار دهید اما امکان بزرگ کردن آنها را فراهم نمایید:** در صورتی که تصاویر با اندازه‌ی بزرگ در صفحه قرار گیرند، زمان لود شدن صفحه را تا حد زیادی افزایش می‌دهند. روش بهتر آن است که به کاربر امکان دهیم تصاویر را در صورت تمایل در حالت بزرگتر ببینند.
- **توجه به سادگی:** در صورتی که با حذف یک عنصر طراحی، برنامه کاربردی با مشکلی مواجه نمی‌شود آن را حذف کنید. زیرا سادگی همواره به پیچیدگی غلبه دارد!



# AESTHETIC DESIGN

❖ علاوه بر صفحه‌بندی سایر مواردی که باید در طراحی جنبه‌های زیبایی صفحه وب مورد توجه قرار گیرند عبارتند از:

➤ رنگ بندی

➤ فونت و style های به کار رفته در متن

➤ استفاده از عکس، صدا، ویدئو، انیمیشن و ...

# طراحی محتوا

❖ فرآیند تولید سیستم‌های مبتنی بر وب متشکل از فعالیتهای موجود در شکل زیر است. این فعالیت‌ها به صورت **iterative و incremental** انجام می‌پذیرند.

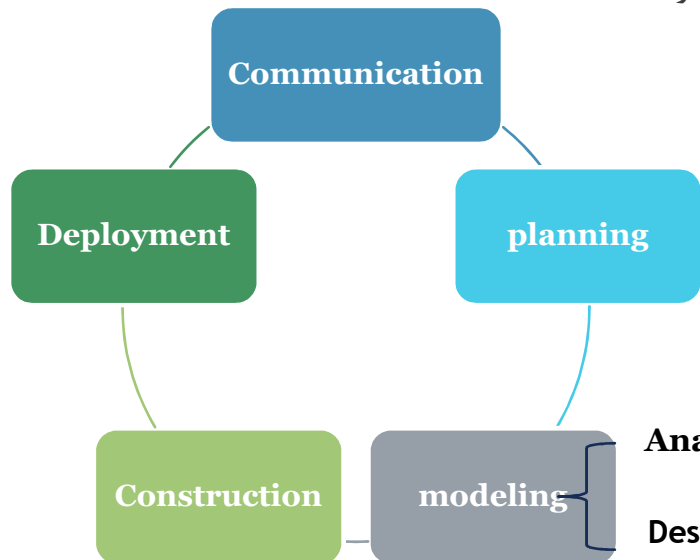
❖ طراحی محتوا در مرحله آنالیز شروع می‌شود و در مرحله طراحی تکامل می‌یابد.

❖ در مرحله آنالیز مدل محتوای ارائه شده شامل تمام عناصر **ساختاری** است که دید مناسبی از نیازمندی‌های وابسته به محتوا ارائه می‌دهند. این مدل شامل **content object** ها (مانند متن، تصاویر گرافیکی، ویدئو، صوت و ...) و سایر کلاس‌های آنالیز (مانند موجودیت‌های خارجی، نقش‌ها، رویدادها، اشیا و ... هستند) است.

❖ رابطه بین **content object** های شناسایی شده در مرحله آنالیز با **content object** هایی که در مرحله طراحی ارائه می‌شوند مانند رابطه‌ی بین کلاس‌های اولیه آنالیز با کلاس‌ها و مولفه‌های سطح طراحی در نرم‌افزارهای غیر وب می‌باشد.

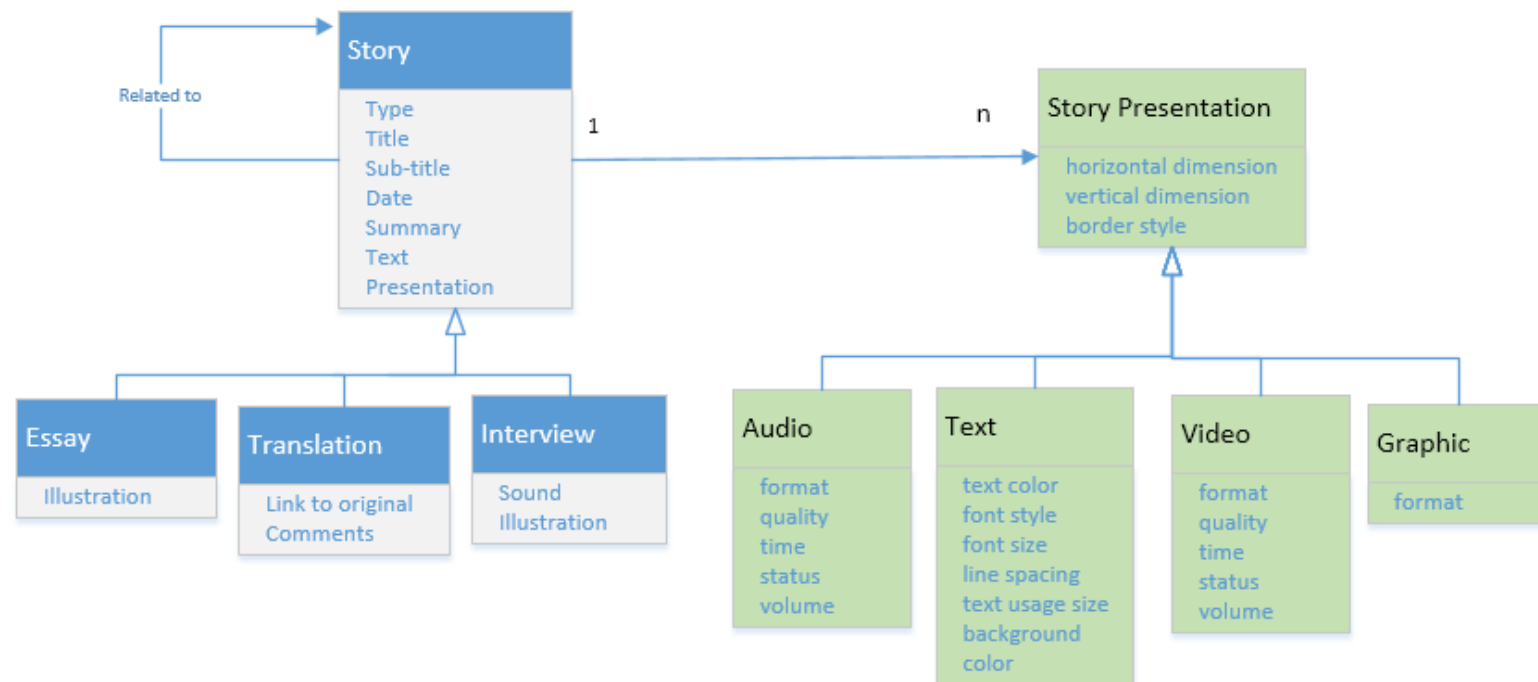
❖ برخی از **attribute** هایی که در **content object** ها قرار می‌گیرند، **attribute** هایی هستند که مستقل از جزئیات طراحی و پیاده‌سازی، تنها خصوصیات محتوا را بیان می‌کنند. این نوع از **attribute** ها معمولاً در مرحله آنالیز شناسایی می‌شوند. اما سایر **attribute** های وابسته به طراحی و پیاده‌سازی در مرحله طراحی **Design** به **content object** ها اضافه می‌گردند.

26



# طراحی محتوا

❖ برای مثال در رابطه با سیستم مجله الکترونیکی، کلاس‌های آنالیز موجود در مدل محتوا با رنگ آبی نشان داده شده‌اند. در طراحی محتوا جزئیات بیشتری به این مدل افزوده می‌شود. این موارد که با رنگ سبز نشان داده شده‌اند content object هایی را نشان می‌دهند که به صفت presentation در کلاس story جزئیات بیشتری را می‌افزاید. (مشخص میکند یک story به چه نحوی ارائه میشود، برای مثال از طریق ویدئو)



27

# طراحی معماری

❖ به صورت کلی در طراحی معماری مولفه‌ها، ارتباط بین آن‌ها و قواعد حاکم بر این ارتباطات تعیین می‌گردد.

❖ طراحی معماری برنامه کاربردی تحت وب شامل دو فعالیت زیر می‌باشد:

➤ طراحی معماری محتوا: بر نحوه ساختاردهی content object ها (همچنین صفحات وب که شامل ترکیبی از این object ها هستند) به منظور امکان‌پذیر ساختن ارائه آن‌ها به کاربر و پیمایش آن‌ها توسط وی تمرکز دارد.

➤ طراحی معماری برنامه کاربردی تحت وب: بر نحوه ساختاردهی برنامه کاربردی تحت جهت مدیریت تعاملات کاربر، پردازش اطلاعات، کنترل پیمایش و ارائه محتوا به کاربر تمرکز دارد.

❖ طراحی معماری برنامه کاربردی تحت وب را می‌توان در موازات طراحی واسط کاربری، طراحی Aesthetic و طراحی محتوا انجام داد.

# معماری محتوا

## ❖ ساختار خطی

➤ این گونه از معماری‌ها زمانی مورد استفاده قرار می‌گیرند که یک دنباله‌ی قابل پیش‌بینی از تعاملات وجود داشته باشد. برای مثال:

- زمانی که کاربر از طریق یک دنباله از گام‌های تعیین شده فعالیت به خصوصی را انجام دهد (مثلا ثبت نام در آزمون سراسری)
- ارائه‌ی یک tutorial به کاربر که در آن صفحات که شامل اطلاعات، تصاویر و .. هستند با یک توالی مشخص به کاربر ارائه می‌شوند. ( هر صفحه در یک دنباله مشخص، تنها پس از این که پیش نیازهای آن مشاهده شده باشند به کاربر ارائه می‌شود).
- فرآیند سفارش محصول نیز به صورت خطی است.

➤ ساده ترین نوع این ساختار، ساختار کاملاً خطی (pure linear) است.

- زمانی که ارائه‌ی محتوا ساختار کاملاً خطی دارد و می‌خواهیم مطمئن شویم که کاربر محتوا را دقیقاً به همان ترتیب موردنظر دریافت کرده است.

✓ یکی از محاسن این ساختار در این است که با توجه به این که گام بعدی کاملاً مشخص است، می‌توان اطلاعات موردنیاز در گام بعد را preload کرد، به این ترتیب کارایی افزایش می‌یابد. ( برای مثال زمانی که کاربر در حال مطالعه‌ی محتوای یک صفحه است، می‌توان تصاویر مورد نیاز در صفحه بعد را load کرد).

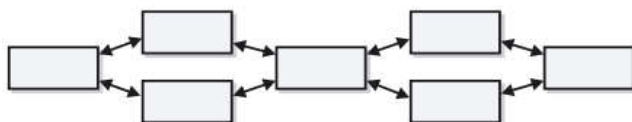
✗ کاربر در معماری کاملاً خطی تنها می‌تواند به صفحه بعدی و قبلی برود، بنابراین زمانی که پیچیدگی محتوای سایت افزایش می‌یابد، این معماری محدودیت‌زا می‌شود.



# معماری محتوا

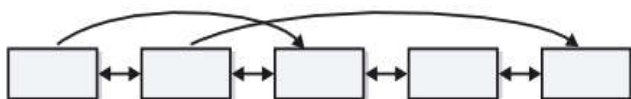
## ❖ ساختار خطی (ساختار خطی دارای alternative)

- در این ساختار همانطور که در شکل مشخص است، کاربر مانند ساختار خطی معمولی یک مسیر متوالی را طی می کند با این تفاوت که ممکن است برای رسیدن به یک از صفحه از صفحه دیگر، چندین مسیر جایگزین وجود داشته باشد.
- برای مثال در یک سایت آزمون، تعدادی سوال وجود دارد که به ترتیب به کاربر نمایش داده می شود. همچنین می خواهیم کاربر پس از این که نتیجه پاسخ خود به هر سوال را دید به سوال بعد برود. به این ترتیب پس از مشاهده صفحه سوال اول، در صورتی که جواب درست باشد، به صفحه ای منتقل می شود که در آن جواب وی تایید می شود؛ اما اگر جواب نادرست داده باشد به صفحه ای می رود که جواب درست را می بیند. نهایتاً صرف نظر از اینکه کاربر جواب درست داده یا نه به صفحه ای منتقل می شود که مربوط به سوال دوم است.
- ✓ اگر چه صفحات در این حالت کاملاً static هستند، اما چون کاربر در هنگام خروج از یک صفحه با توجه به سناریوهای موجود با مسیرهای جایگزین متفاوت روبروست، تا حدی تصور می کند که صفحات حالت پویا دارند.



## ❖ ساختار خطی (ساختار خطی و optional)

- این نوع از معماری برای زمانی مناسب است که به صورت کلی یک دنباله متوالی از صفحات وجود دارد، با این وجود ممکن است در این دنباله از تعاملات تا حدی تنوع و گوناگونی وجود داشته باشد. (برای مثال بعضی از صفحات skip شوند).



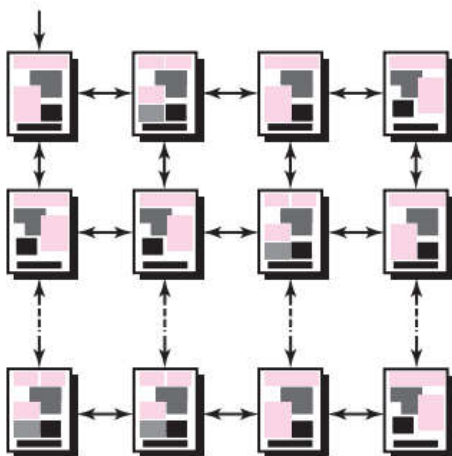
# معماری محتوا

## ❖ ساختار grid

➤ این نوع از معماری زمانی مورد استفاده قرار می‌گیرد که محتوای برنامه کاربردی را بر مبنای دو یا چند بعد مختلف می‌توان دسته‌بندی نمود.

■ برای مثال در یک وب سایت فروش لوازم دیجیتال، اقلام را می‌توان بر اساس نوع آن‌ها دسته‌بندی نمود. (مثلا موبایل، حافظه، لوازم جانبی و ...) همچنین می‌توان این اقلام را در بعد دیگر براساس سازنده آن‌ها دسته‌بندی کرد. با استفاده از ساختار grid، می‌توان برای کاربران این امکان را فراهم کرد که در عین حال که در بعد افقی به دنبال یکی از اقلام می‌گردد، همزمان سازنده‌های مختلف این محصول را نیز بررسی کند.

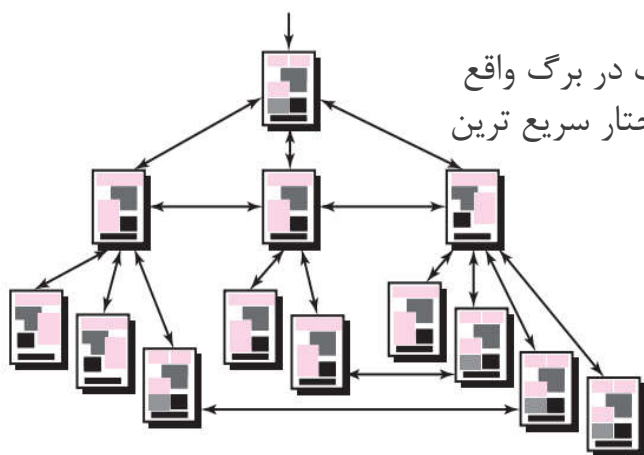
✓ این ساختار زمانی قابل استفاده است که محتوا دارای ساختار منظم و مشخصی باشد.



# معماری محتوا

## ❖ ساختار سلسله مراتبی

- مرسوم ترین نوع معماری در بین برنامه‌های کاربردی تحت وب است.
- اگر چه ساختار سلسله‌مراتبی نمی‌تواند ساختار منظمی مانند grid و یا یک ساختار قابل پیش‌بینی مانند ساختار خطی فراهم کند، اما در عوض با این ساختار می‌توان abstraction را در ساختار محتوا ایجاد نمود. (در سطوح بالاتر سلسله مراتب، جزئیات را از دید کاربر مخفی کرد و در سطوح پایین‌تر جزئیات بیشتری را در اختیار او قرار داد).
- در بالاترین سطح این سلسله مراتب صفحه اصلی سایت قرار دارد. از این صفحه می‌توان گزینه‌های مختلفی را انتخاب کرد. وقتی که کاربر یکی از این گزینه‌ها را انتخاب کرد وارد سطح بعدی سلسله‌مراتب می‌شود و همین طور با انتخاب‌هایی که انجام می‌دهد، عمق بیشتری از سلسله مراتب را طی می‌کند تا نهایتاً به صفحه‌ای مورد نظر یا صفحه‌ای که در برگ واقع است برسد.



- در این نوع از معماری، برای اینکه کاربر به یک صفحه که در این سلسله مراتب در برگ واقع است، برسد باید انتخاب متعددی انجام بدهد، اما در برخی از سایت‌ها این ساختار سریع‌ترین روش برای رساندن کاربر به دسته‌بندی مورد نظر او می‌باشد.



# معماری محتوا

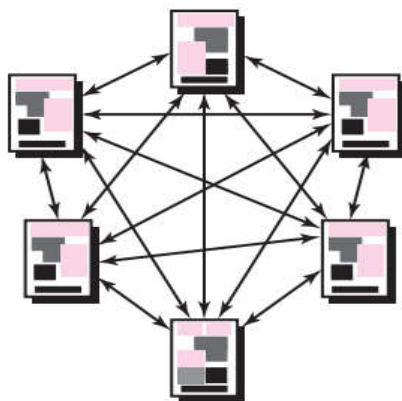
## ❖ ساختار شبکه‌ای

➤ در این ساختار هر صفحه می‌تواند به صفحه دیگر مرتبط باشد.

➤ انعطاف پذیری زیادی در پیمایش فراهم می‌آورد اما در عین حال ممکن است موجب سردرگمی کاربر شود.

✗ در یک ساختار شبکه‌ای، در صورتی که تمام صفحات به هم دیگر مرتبط باشند، تعداد لینک‌ها به میزان زیادی افزایش می‌یابد. به عبارتی در صورتی که  $p$  صفحه داشته باشیم؛  $p(p-1)$  لینک خواهیم داشت که تمام صفحات را به هم مرتبط میکنند. بنابراین در عمل معمولاً ساختار شبکه ای به شکلی استفاده می شود که در آن تنها برخی از صفحات که به یکدیگر مرتبطند، به هم لینک دارند.

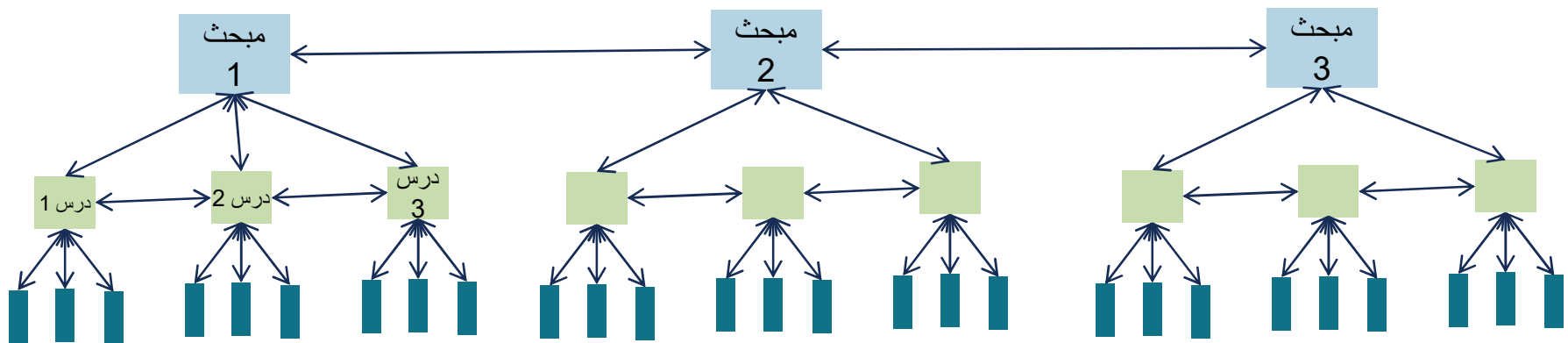
■ برای مثال سایت ویکی پدیا از ساختار شبکه ای استفاده می‌کند. زیرا در هر صفحه، می‌توان از طریق لینک‌هایی که در مفاهیم و موضوعات آن صفحه وجود به سایر صفحات رفت. (cross referencing بین صفحات)



# معماری محتوا

❖ در یک برنامه کاربردی تحت وب می‌توان از چندین ساختار برای محتوا استفاده کرد. ( ترکیبی از ساختارهای معرفی شده)

➤ برای مثال در یک سیستم آموزش دروس، ساختار محتوا در بالاترین سطح به صورت خطی طراحی شده است. به این ترتیب کاربران ابتدا باید مبحث اول، سپس به مبحث ۲ و به همین ترتیب سایر مباحث را بررسی نمایند. در سطح پایین‌تر، ساختار تعیین شده برای ارائه محتوای مبحث اول به صورت سلسله مراتبی انتخاب شده است. به این ترتیب هر مبحث شامل تعدادی درس بوده، هر درس شامل تعدادی مثال و تعدادی آزمون و تعدادی تمرین می‌باشد.



# معماری WEBAPP

❖ معماری Model-View-Controller (MVC) یکی از معماری هایی است که interface را از functionality و محتوای اطلاعاتی برنامه کاربردی تحت وب مجزا می سازد. معماری MVC سیستم را به سه مولفه تقسیم می کند:

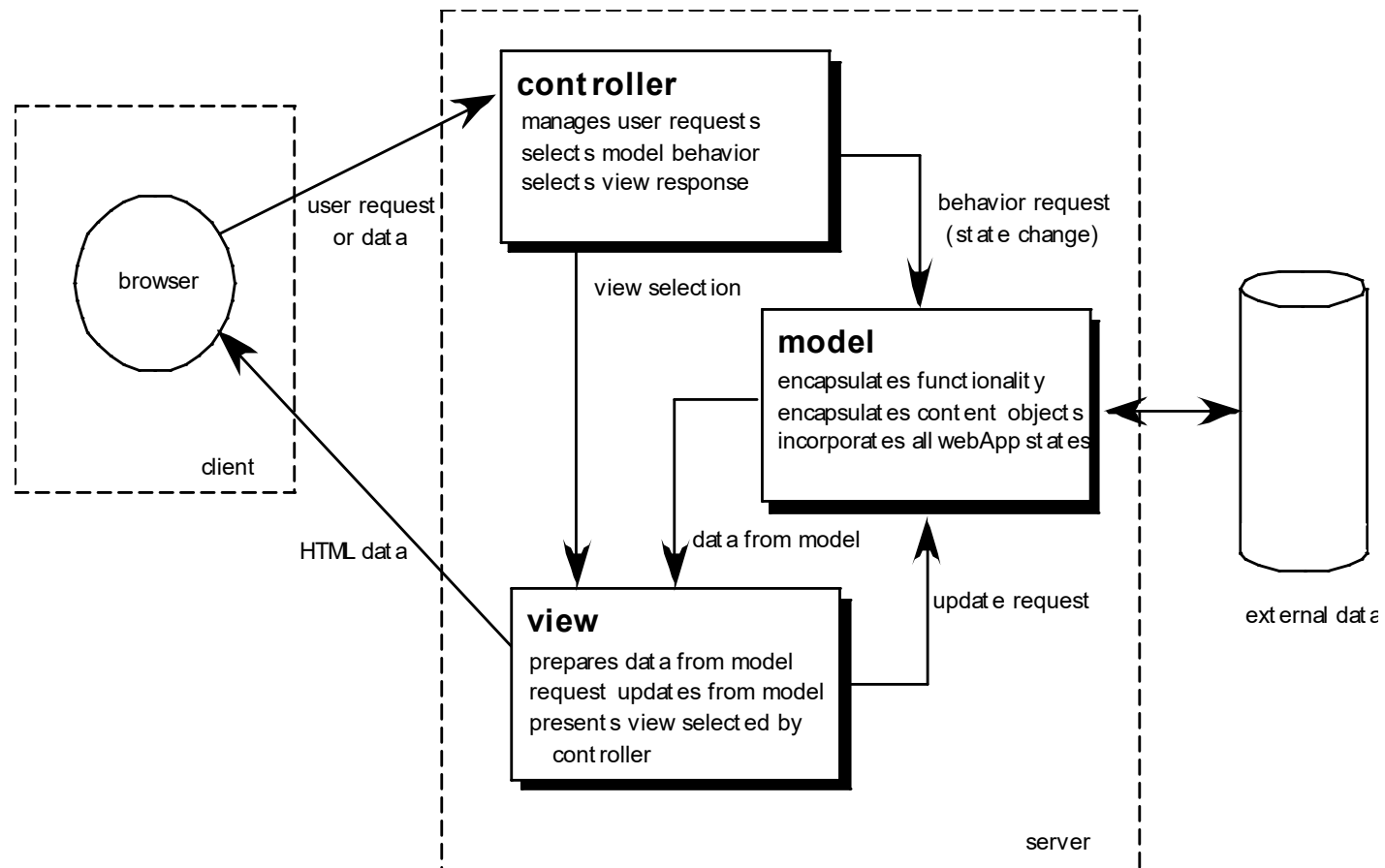
➤ **Model**: بخش Model شامل داده ها و منطق پردازشی (کارکردهای) یک application است. به عبارتی مدل تمام content object، دسترسی به منابع اطلاعاتی ( internal یا external ) و تمام functionality های اصلی یک application را در اختیار قرار می دهد.

➤ **View**: شامل تمام interface هایی است که اطلاعات را به کاربر نشان می دهند.

➤ **Controller**: دسترسی به model و view را مدیریت کرده و امکان انتقال داده بین این دو را فراهم می کند.

✓ در معماری MVC، **view** توسط **controller** با توجه به ورودی ها و **event** های دریافت شده از کاربر از طریق دریافت داده های موجود در **model**، **update** میشود.

# معماری WEBAPP



## طراحی پیمایشی

❖ در طراحی معماری برنامه کاربردی وب، مولفه‌ها (صفحات، اپلت‌ها، اسکریپت‌ها، وب‌سروها، دیتا سروورها و ...) شناسایی می‌شوند. در طراحی پیمایشی به دنبال هستیم که مشخص کنیم کاربر چگونه با استفاده از مسیرهای پیمایش معین می‌تواند به این مولفه‌ها دسترسی داشته باشد و بین آن‌ها پیمایش کند.

❖ برای این منظور در طراحی پیمایشی باید مشخص کنیم که:

➤ کاربر برای دستیابی به نیازمندی‌های خود به چه مولفه‌هایی، با چه ترتیب و در چه توالی مشخصی دسترسی دارد.  
(Navigation semantic)

➤ همچنین باید معین کنیم، که ارتباط بین مولفه‌ها به چه نحوی انجام می‌شود (Navigation syntax) (برای مثال از طریق لینک‌های موجود در متن، از طریق منوها، site map و ...)

### ❖ Navigation semantic

➤ طراحی Navigation semantic براساس actorها و use case های شناسایی شده برای آن‌ها انجام می‌پذیرد. (با توجه به اینکه هر actor بر مبنای نیازمندی‌های خود به صورت متفاوتی برنامه کاربردی تحت وب را پیمایش می‌نماید).

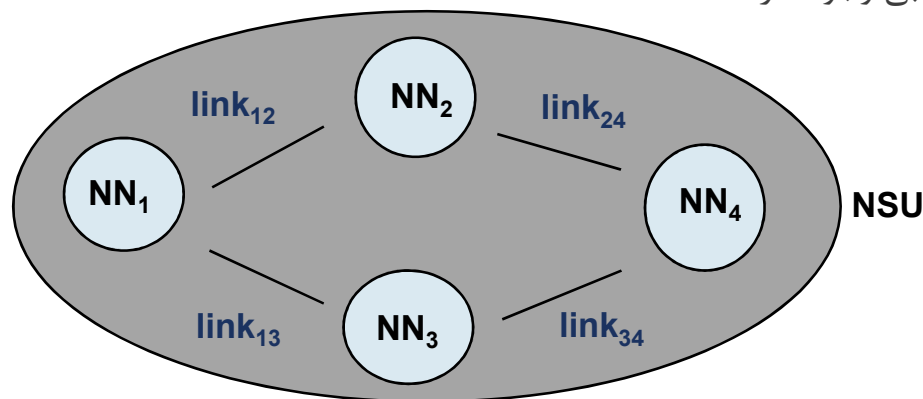
➤ بر مبنای use case های تعیین شده در رابطه با هر actor می‌توان مشخص کرد که هر actor به چه content object ها و کارکردهایی دسترسی دارد.

37

# طراحی پیمایشی

## ❖ Navigation semantic (ادامه)

- هر actor در هنگام تعامل با برنامه کاربردی تحت وب، با یک مجموعه از (NSU) navigation semantic unit ها در ارتباط است. هر NSU مشخص کننده این است که یک actor در رابطه با یک usecase چگونه بین content object های مختلف پیمایش می کند و برای این منظور چه نیازمندی هایی از نظر پیمایش دارد.
- برای این منظور هر NSU شامل مجموعه عناصری است که (WON) way of navigation نام دارند. هر WON متشکل از تعدادی (NN) navigational node است که از طریق لینک هایی به یکدیگر مرتبط شده اند. WON بهترین مسیر پیمایش بین این content object ها را برای دستیابی به نیازمندی مشخصی توسط یک actor به خصوص معلوم می نماید.
- یک navigational link ممکن است navigational node را به یک NSU مرتبط کند. بنابراین امکان به کارگیری NSU ها در یک ساختار سلسله مراتبی وجود دارد.



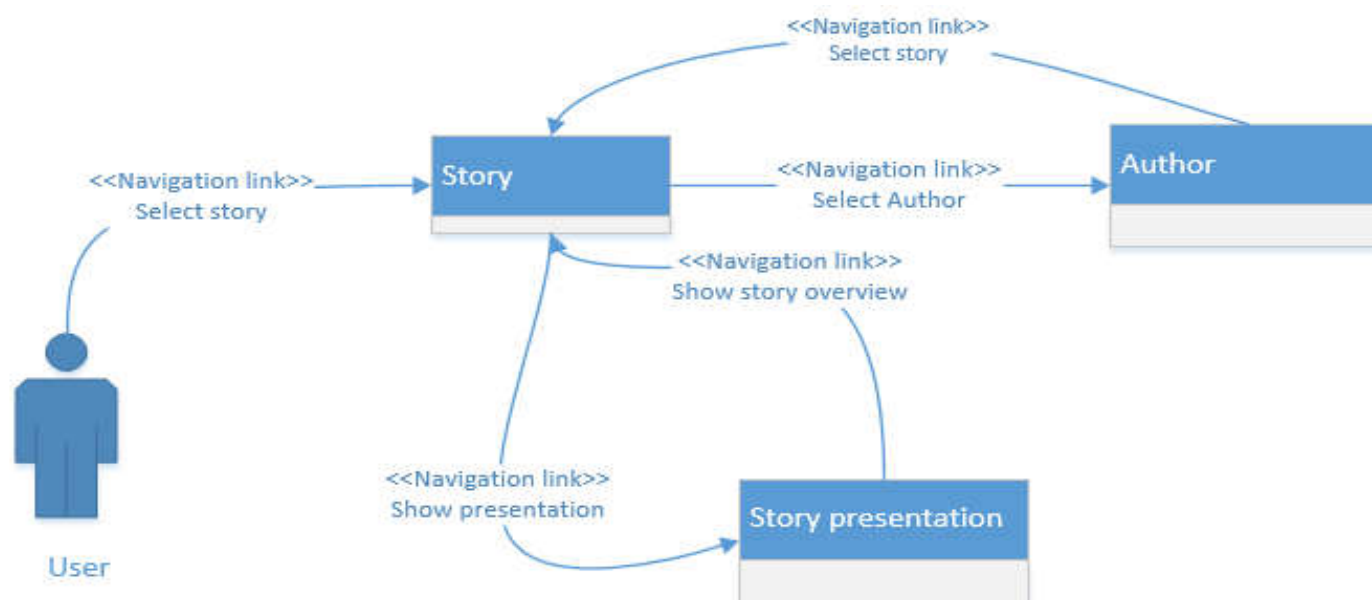
# طراحی پیمایشی

## ❖ Navigation semantic (ادامه)

➤ برای مثال شرح یکی از usecase های سیستم مجله آنلاین به صورت زیر است:

« کاربر باید بتواند اطلاعات یک story را مشاهده کند در صورت تمایل به presentation آن دسترسی داشته باشد. همچنین در صورتی که کاربر در حال مشاهده یک story باشد که توسط نویسنده یا نویسندگانی نوشته شده است، باید این امکان وجود داشته باشد که کاربر سایر story های نوشته شده توسط هر یک از نویسندگان را مشاهده کند. »

❖ NSU این usecase در شکل زیر رسم شده است.



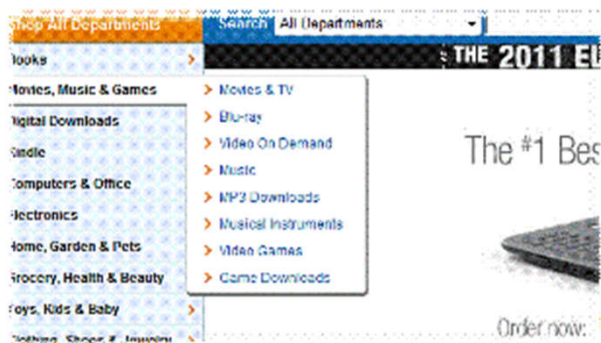
39

# طراحی پیمایشی

## Navigation syntax ❖

➤ نحوه‌ی انجام navigation بین Navigation node ها را مشخص می‌کند و می‌تواند به یکی از صورت‌های زیر باشد:

- Individual navigation link: شامل لینک‌های متنی، آیکن‌ها، دکمه‌ها، تصاویر گرافیکی و مواردی از این دست می‌باشد. این گونه لینک‌ها باید محتوا مطابقت داشته باشند و از لحاظ شهودی با انتظارات کاربر مطابقت داشته باشند. (برای مثال برای submit یک فرم از دکمه استفاده شود نه لینک متنی)
- Horizontal navigation bar: دسته‌بندی‌های اصلی برای محتوا و کارکردهای موجود را به صورت افقی لیست می‌کند. معمولاً شامل ۴ تا ۷ دسته اصلی می‌باشد.
- Vertical navigation column (۱): دسته‌بندی‌های اصلی برای محتوا و کارکردهای موجود را به صورت عمودی لیست می‌کند. (۲) در داخل دسته‌بندی‌های دیگر استفاده می‌شود. برای مثال وقتی روی یک دسته اصلی کلیک می‌شود دسته‌بندی‌های فرعی آن به صورت عمودی نمایش داده می‌شوند.



40



# طراحی پیمایشی

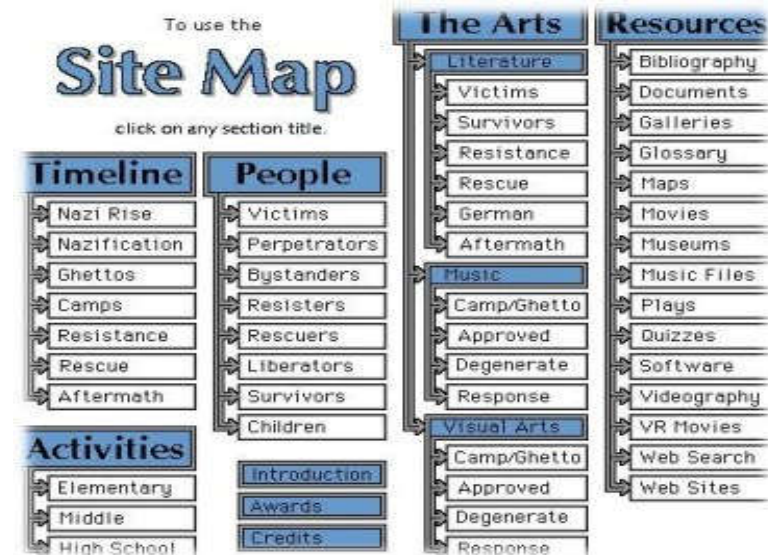
## Navigation syntax ❖

➤ نحوه‌ی انجام navigation بین Navigation node ها را مشخص می‌کند و می‌تواند به یکی از صورت‌های زیر باشد:

▪ Tabs:

▪ Site maps: یک جدول در برگیرنده‌ی همه‌ی لینک‌ها

برای دسترسی به هر جایی از web app



41

## الگوهای طراحی در برنامه های کاربردی تحت وب

❖ الگوهای طراحی برنامه های کاربردی تحت وب براساس دو مورد زیر دسته بندی می شوند:

➤ design focus: مشخص می کند که یک الگو بر کدام بعد از طراحی برنامه ی کاربردی تحت وب تمرکز دارد ( مثلا طراحی واسط کاربری، طراحی معماری، طراحی پیمایشی)

➤ level of granularity: مشخص می کند که یک الگو در چه سطحی بر برنامه ی کاربردی تحت وب قابل اعمال است. ( مثلا بر تمام برنامه کاربردی، بر یک زیر سیستم، یک صفحه وب و ....)

❖ الگوهای برنامه های کاربردی تحت وب را می توان در دسته های زیر تقسیم بندی نمود:

➤ الگوهای interaction

➤ الگوهای navigation

➤ الگوهای Content

➤ الگوهای Architecture

## الگوهای طراحی در برنامه های کاربردی تحت وب

### الگوهای NAVIGATION

#### نام الگو: Directory Navigation

**problem** کاربر می خواهد یک قلم (item) را از میان مجموعه بزرگی از اقلام انتخاب نماید.

اطلاعات level 1 (مجموعه اقلام) و level 2 (قلم موردنظر) را در کنار هم قرار دهید. به عبارتی یک category اصلی خواهیم داشت که در زیر آن چند sub-category اش را مشخص کردیم. به این ترتیب کاربر می تواند چند مورد از sub category های هر category را به طور همزمان مشاهده کند و راحت تر تشخیص دهد که قلم موردنظر در کدام category قرار داد. بنابراین روی این category کلیک می کند تا تمام subcategory های آن را در یک صفحه مجزا مشاهده نماید.

#### Solution

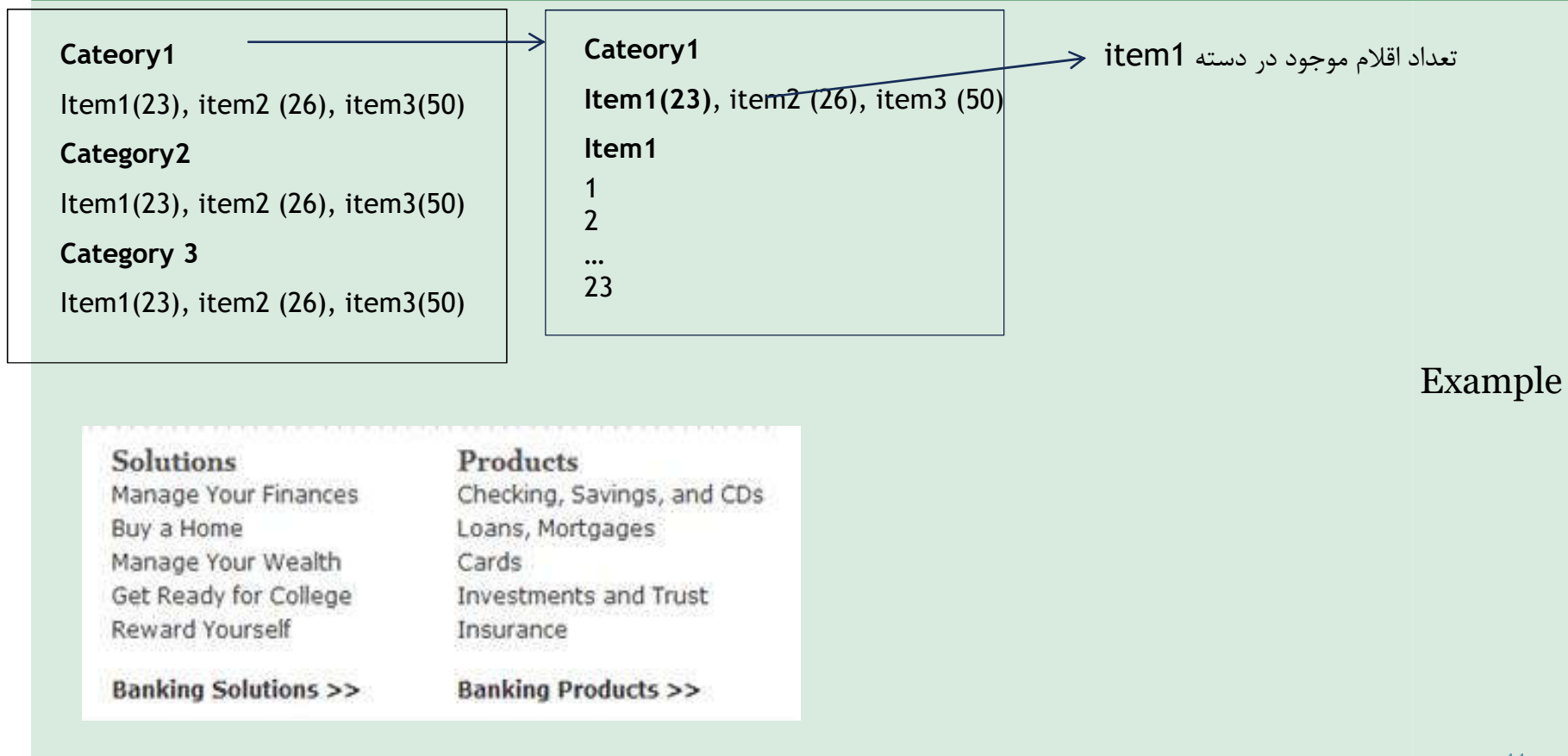
زمانی استفاده می شود که چندین مجموعه بزرگ از اقلام داریم. کاربر می خواهد قادر باشد بین اقلامی که در یک مجموعه قرار دارند به راحتی حرکت کند. همچنین کاربر می خواهد یک نمای کلی از اقلام موجود در مجموعه های مختلف را مشاهده نماید.

#### When used

# الگوهای طراحی در برنامه های کاربردی تحت وب

## الگوهای NAVIGATION

### نام الگو: Directory Navigation



## الگوهای طراحی در برنامه های کاربردی تحت وب

### الگوهای NAVIGATION

#### نام الگو: News

##### problem

در یک سیستم تحت وب پویا، مانند یک سیستم فروشگاهی که محتوا (محصولات) و کارکردهای ارائه شده آن به سرعت در حال گسترش است، می‌خواهیم کاربر همواره در جریان اقلامی که به طور پیوسته به سیستم افزوده می‌شود، قرار گیرد.

##### Solution

فضای صفحه اصلی (home page) را طوری ساختاردهی نمایید که یک بخش از آن به اقلامی که تازه به سایت اضافه شده است تعلق داشته باشد. در این بخش همواره خلاصه‌ای از اقلامی که به تازگی به سایت اضافه شده‌اند؛ به همراه لینکی برای دسترسی به این اقلام وجود دارد. به این ترتیب کاربر مجبور نیست، مسیر معمولی تعیین شده تا رسیدن به اقلام را پیمایش کند و با سرعت بیشتری به اقلام جدید دست پیدا کند. البته باید دقت کرد که لینک های موقتی که برای این بخش ایجاد می‌شوند باید مدیریت شوند (برای مثال در صورت به روز رسانی اقلام موجود در این بخش، لینک های قدیمی‌تر باید غیرفعال شده و از کار بیفتند).

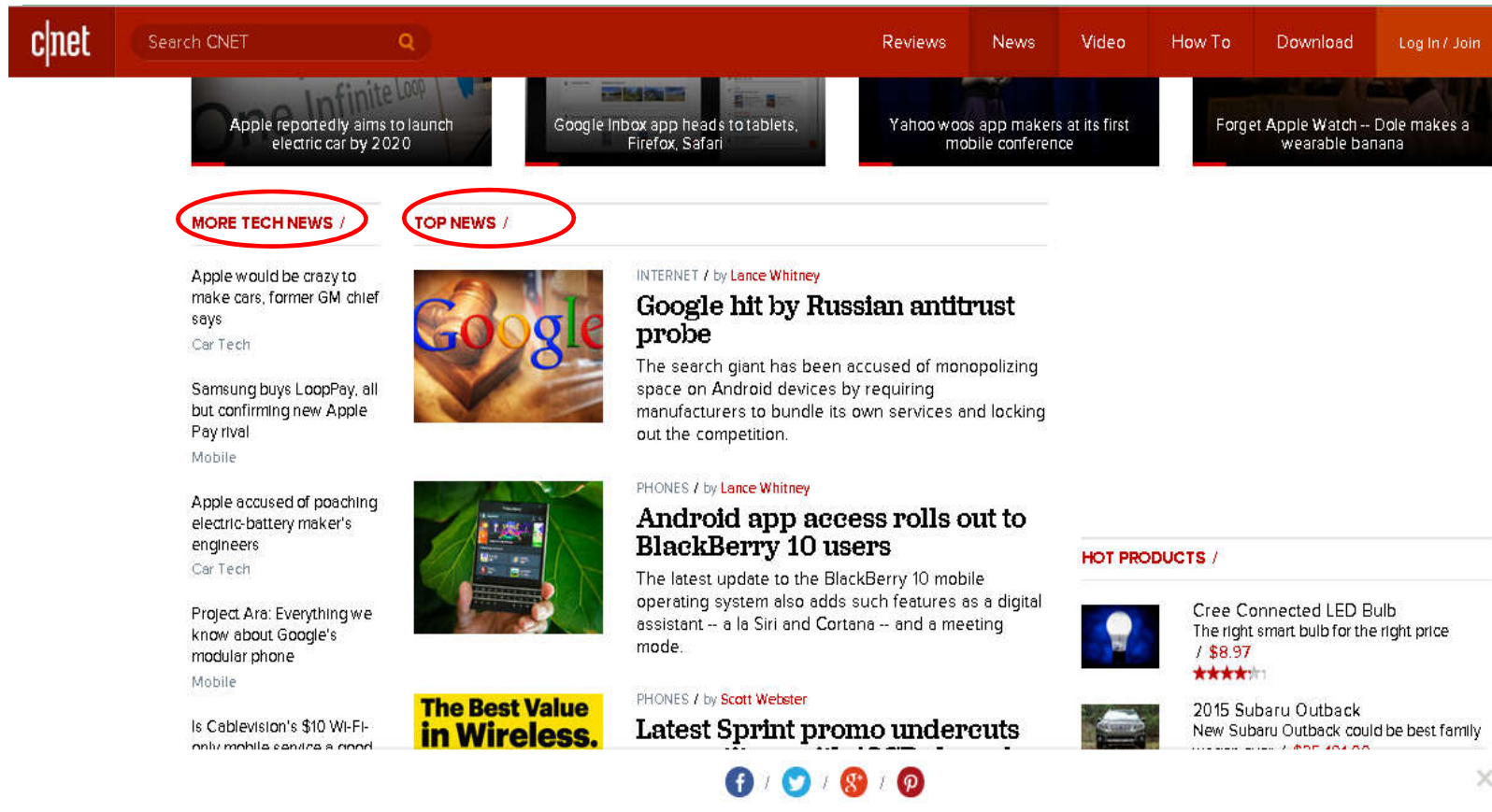
##### When used

زمانی که آگاهی کاربران از محتوای جدید افزوده شده به سایت دارای اهمیت زیادی است.

# الگوهای طراحی در برنامه های کاربردی تحت وب

## الگوهای NAVIGATION

نام الگو: News



Example

<http://www.news.com>

46

## الگوهای طراحی در برنامه های کاربردی تحت وب

### الگوهای NAVIGATION

#### نام الگو: Clear entry point

**problem** برنامه کاربردی تحت وب دارای محتوا و کارکردها متنوع و زیادی است که مشخص نیست کاربران به چه نحوی از آن استفاده خواهند کرد. در صورتی که تمام این محتوا و کارکرد به صورت یکباره در اختیار کاربر قرار گیرند موجب سردرگمی کاربر می شود.

**Solution** تعداد کمی از کارکردهای اصلی که اکثر کاربران سایت از آن استفاده میکنند، در صفحه و منوهای اصلی ارائه می گردد. بنابراین این کارکردهای اصلی مانند **entry point** هایی هستند که به سایر کارکردهای متنوع و فرعی منجر می شوند. در صورتی که کاربر یکی از کارکردهای اصلی و **general** را انتخاب کند، به تدریج از طریق **entry point** هایی که به میزان بیشتری تخصصی هستند به **context** موردنظر خود هدایت می شود. به این ترتیب کاربرانی که با کارکردهای اصلی در ارتباطند وارد کارکردهای جزئی و غیرضروری نمی شوند.

**When used**

- تعدادی **task** و کارکرد اصلی از پیش تعیین شده یا قابل شناسایی وجود دارد که اکثر کاربران از آن استفاده می کنند.
- عمده کاربران سایت افرادی هستند که به دفعات کم یا برای اولین بار به سایت مراجعه می کنند. ( در صورتیکه کارکردها واضح باشند و برنامه کاربردی ساده باشد لزومی به استفاده از این الگو نیست.)

[www.google.com](http://www.google.com)

**Example**

# OOHDM

❖ تاکنون متدلوژی‌های مختلفی برای تولید برنامه‌های کاربردی تحت وب ارائه شده است، مانند:

- Relation management methodology (RMM)
- scenario-based object-oriented hypermedia design methodology (SOHDM)
- Object-Oriented Hypermedia Design Method (OOHDM)
- Web Modeling Language(WebML)

## ❖ OOHDM

- از پرکاربردترین متدلوژی‌های ارائه شده برای تولید برنامه‌های کاربردی تحت وب است.
- مبتنی بر مدل است. ( artifact های تولید شده در مراحل آن مدل هستند.)
- تمرکز زیادی بر روی طراحی واسط گرافیکی و ساختار پیمایشی دارد.
- مراحل فرآیند آن عبارتند از: ( دارای یک فرآیند incremental, iterative و مبتنی بر prototype است.)

توضیحات هر یک از گام‌ها در ادامه بر مبنای سیستم فرضی مجله آنلاین ارائه می‌گردد.

- طراحی مفهومی (Conceptual design)
- طراحی پیمایشی (Navigational design)
- طراحی واسط انتزاعی (abstract interface design)
- پیاده‌سازی (Implementation)



# OOHDM

## ❖ طراحی مفهومی (Conceptual design)

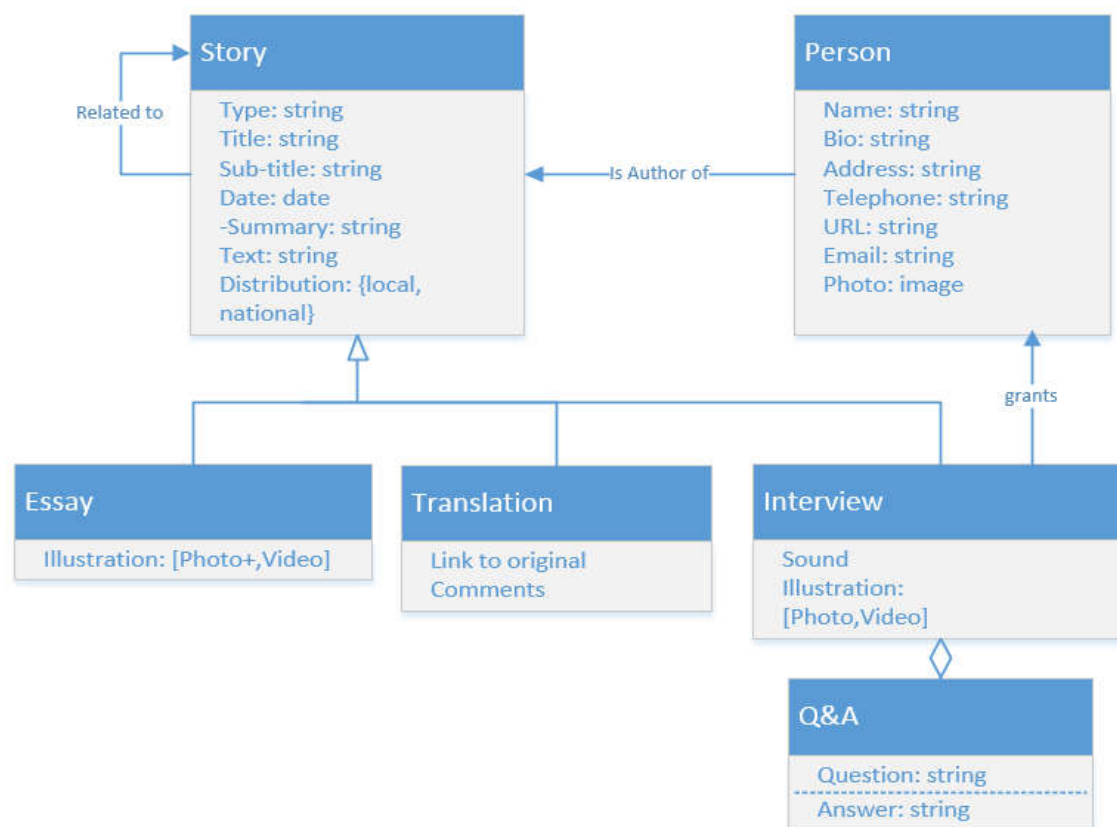
- هدف از طراحی مفهومی، ارائه‌ی یک مدل از دامنه‌ی یک برنامه‌ی کاربردی تحت وب بر مبنای اصول شی‌گرایی است.
- خروجی این گام، یک شمای کلاس است که متشکل از زیر سیستم‌ها، کلاس‌ها، صفات و ارتباط بین آن‌ها می‌باشد.
- Concern اصلی این شمای مفهومی، مدل‌سازی semantic و زیرساخت اطلاعاتی دامنه‌ی یک برنامه کاربردی تحت وب است که شامل مفاهیم دامنه و ارتباطات بین آن‌ها می‌باشد. (بدون توجه به نحوه و قالب ارائه‌ی این اطلاعات و صرف نظر از در نظر گرفتن نوع کاربران و task هایی که آن‌ها می‌توانند بر روی این شمای مفهومی انجام دهند).
- در ارائه‌ی شمای مفهومی اصول طراحی شی‌گرا مانند abstraction, composition, aggregation, generalization و specialization مورد توجه هستند.
- زبان مورد استفاده در طراحی شمای مفهومی، بر مبنای UML است. البته برخی از علائم به کار رفته در طراحی این شما در UML وجود ندارد. (برای مثال برای attribute ها می‌توان چندین type را در نظر گرفت، که هر کدام یک perspective مختلف روی یک موجودیت واحد فراهم می‌آورند و اصطلاحاً به آن‌ها attribute perspective گفته می‌شود).

■ برای مثال یک attribute می‌تواند دارای دو type ، audio و video باشد.

49

# OOHDM

## ❖ طراحی مفهومی در سیستم مجله آنلاین



# OOHDM

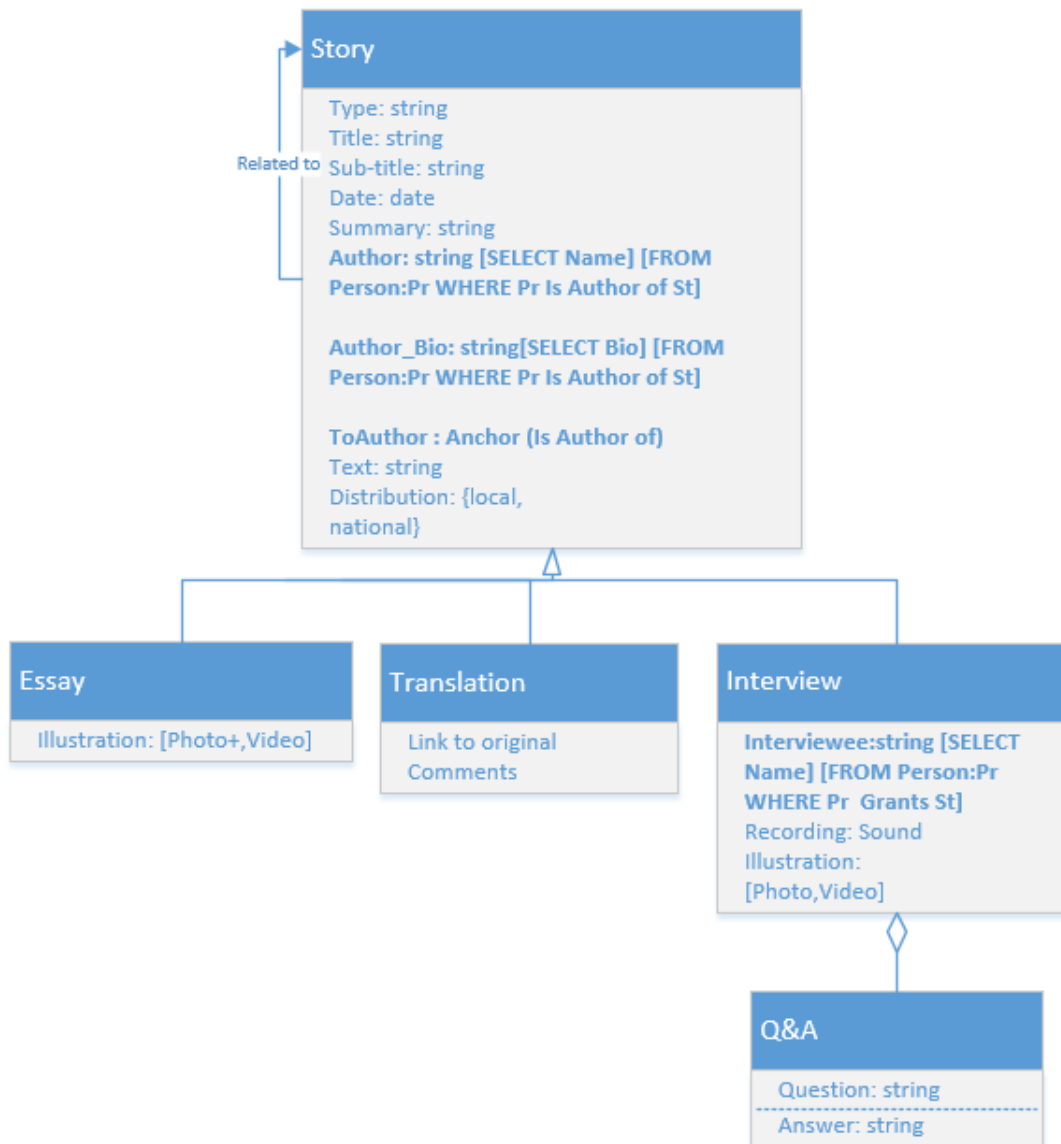
## ❖ طراحی پیمایشی (Navigational design)

- همانطور که گفتیم مدل مفهومی، مفاهیم موجود در دامنه را بدون توجه به نوع کاربران و نحوه استفاده آنها از این اطلاعات مدل می‌کند.
- با این وجود با توجه به اینکه برنامه کاربردی نهایتاً توسط تعدادی کاربر مورد استفاده قرار می‌گیرد که هر یک به نحوی با محتوای موجود در تعامل هستند، بنابراین باید به نحوی اطلاعات بازنمایی شده در مدل مفهومی را سازمان‌دهی کرد که در آن نوع کاربران و استفاده‌هایی که از این داده‌ها می‌برند قابل بیان باشد. این کار از طریق مدل پیمایش انجام می‌پذیرد.
- به این ترتیب مدل پیمایش به صورت یک view بر روی مدل مفهومی تعریف می‌شود. بنابراین می‌توان بر مبنای یک مدل مفهومی، چندین مدل پیمایشی برای کاربران مختلف طراحی کرد.
- طراحی پیمایشی در دو شما انجام می‌پذیرد:
  - شماي کلاس (navigational class schema)
  - شماي زمینه (navigation context schema)

# OOHDM

## ❖ طراحی پیمایشی - شمای کلاس (navigational class schema)

- این شما بیان کننده‌ی کلاسی از object هایی است که در برنامه‌ی کاربردی قابل پیمایش هستند. هر یک از کلاس‌ها به صورت یک view بر روی مدل مفهومی تعریف می‌شوند.
- در OOHDM، Navigation class های مشخصی تعریف شده‌اند که ساختار پیمایش را شکل می‌دهند. این کلاس‌ها عبارتند از:
  - Node ها، لینک‌ها و ساختارهای دسترسی (access structures) (ساختارهای دسترسی مانند اندیس‌ها، guided tours که روش‌های دسترسی به node ها را معین می‌سازند).
  - Node ها به صورت view های شی‌گرا بر روی کلاس‌های مدل مفهومی از طریق زبان query (چیزی شبیه به sql) تعریف می‌شوند. به این ترتیب یک Node ترکیبی از attribute های کلاس‌های مرتبط با هم در مدل مفهومی است.
  - یکی از انواع attribute که در داخل یک node قابل تعریف است، anchor ها هستند. Anchor مشخص کننده‌ی ارتباط بین node ها هستند و لینک‌ها موجود در مدل navigation براساس آنها شکل می‌گیرند. یک anchor به صورت view بر روی ارتباطات بین کلاس‌های موردنظر در مدل مفهومی تعریف می‌شوند.



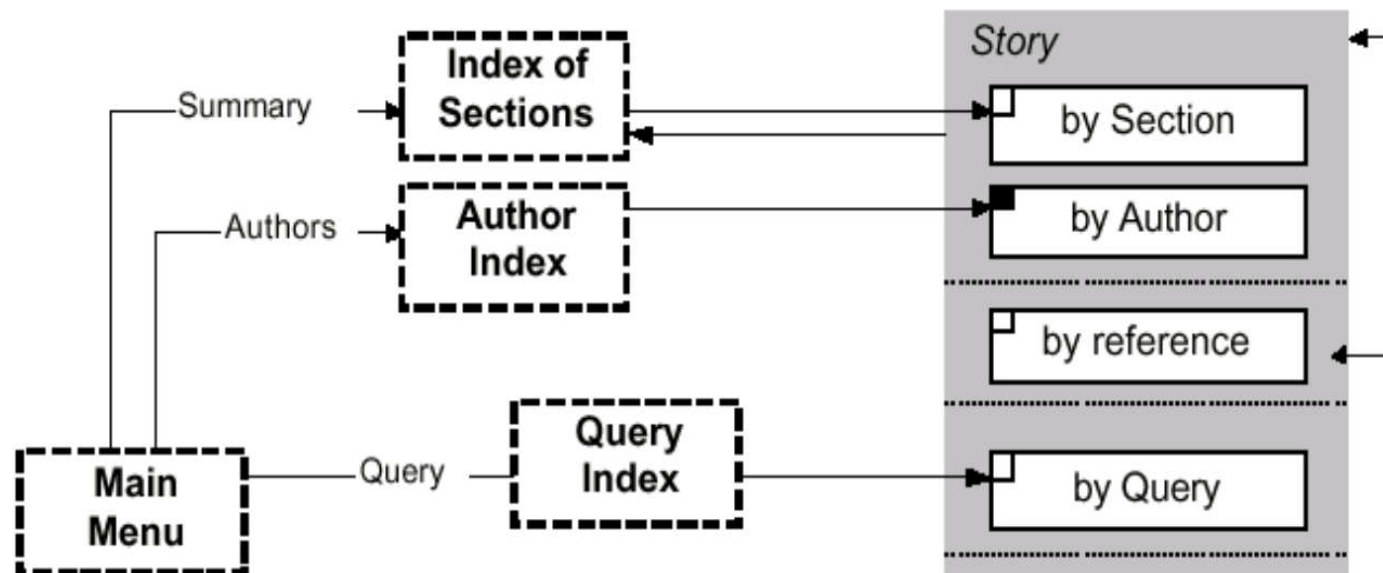
❖ طراحی پیمایشی در سیستم مجله  
آنلاین - شمای کلاس

# OOHDM

## ❖ طراحی پیمایشی - شمای زمینه (navigation context schema)

- در شمای کلاس پیمایشی، کلاس‌های قابل پیمایش و ارتباط بین آن‌ها تعریف شدند. پس از انجام این کار در طراحی شمای زمینه باید مشخص کنیم که کاربران چگونه براساس اهداف مختلفی که دارند با استفاده از Object هایی که از روی کلاس‌های تعریف شده ایجاد می‌گردد در فضای برنامه کاربردی پیمایش را انجام می‌دهند. (مانند تعریف navigation semantic unit و way of navigation)
- زمینه‌ی پیمایش شامل یک سری node، لینک‌ها، کلاس‌های زمینه و سایر شماهای زمینه (امکان به کارگیری شماهای زمینه به صورت تو در تو وجود دارد). می‌باشد.

# OOHDM



# OOHDM

NC	Story By Author
includes	$\forall s : \text{Story}, a : \text{Person}. (a, s) \in \text{IsAuthorOf}$ $\wedge a.\text{name} = \langle \text{author} \rangle$
	entrypoint: e1 [first node]
	path :circular , ordered ascending on a.date
	behavior : step by step



# OOHDM

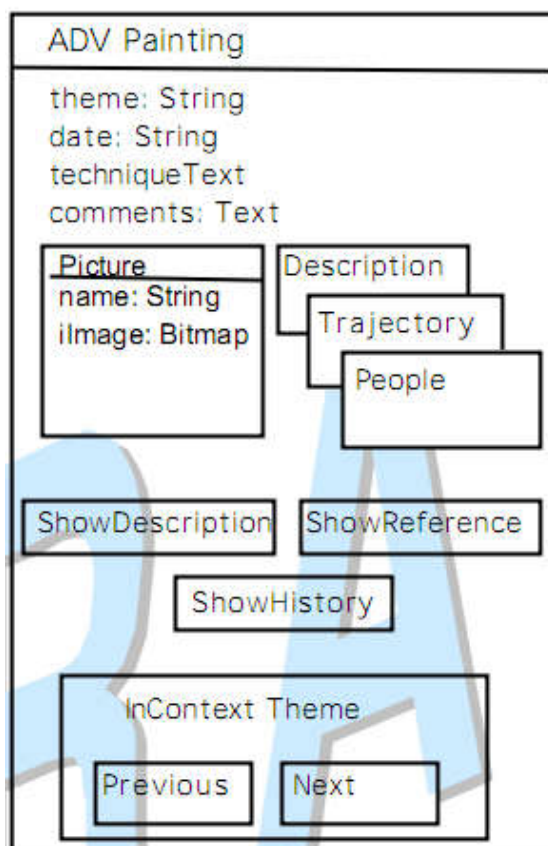
## ❖ طراحی واسط انتزاعی (abstract interface design)

- پس از انجام طراحی پیمایشی، باید واسط کاربری که از طریق آن کاربر طبق مدل پیمایشی طراحی شده به تعامل با برنامه‌ی کاربردی تحت وب می‌پردازد، تعیین شود.
- جداسازی طراحی واسط انتزاعی از طراحی پیمایشی موجب می‌شود که بتوانیم برای یک مدل پیمایشی واحد چندین واسط کاربری بسازیم. (استقلال مدل پیمایش از واسط کاربری)


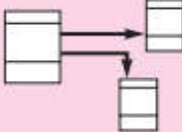


➤ در OOHDM، طراحی واسط انتزاعی از طریق روش طراحی Abstract Data View(ADV) انجام می‌پذیرد.

➤ با استفاده از ADV می‌توان موارد زیر را تعیین نمود:

- ساختار static واسط کاربری را نمایش می‌دهد ( شامل عناصر به کار رفته برای نمایش عناصر موجود در مدل پیمایشی و سایر عناصر گرافیکی مانند منوها، دکمه‌ها )
- ارتباط بین این عناصر static با عناصر مدل پیمایشی
- چگونه عناصر واسط کاربری به رویدادهای خارجی پاسخ می‌دهند.



# OOHDM

	 Conceptual design	 Navigational design	 Abstract interface design	 Implementation
Work products	Classes, subsystems, relationships, attributes	Nodes links, access structures, navigational contexts, navigational transformations	Abstract interface objects, responses to external events, transformations	Executable WebApp
Design mechanisms	Classification, composition, aggregation, generalization specialization	Mapping between conceptual and navigation objects	Mapping between navigation and perceptible objects	Resource provided by target environment
Design concerns	Modeling semantics of the application domain	Takes into account user profile and task. Emphasis on cognitive aspects.	Modeling perceptible objects, implementing chosen metaphors. Describe interface for navigational objects.	Correctness; application performance; completeness

61