Agile Principles(II)

Dr. Elham Mahmoudzadeh

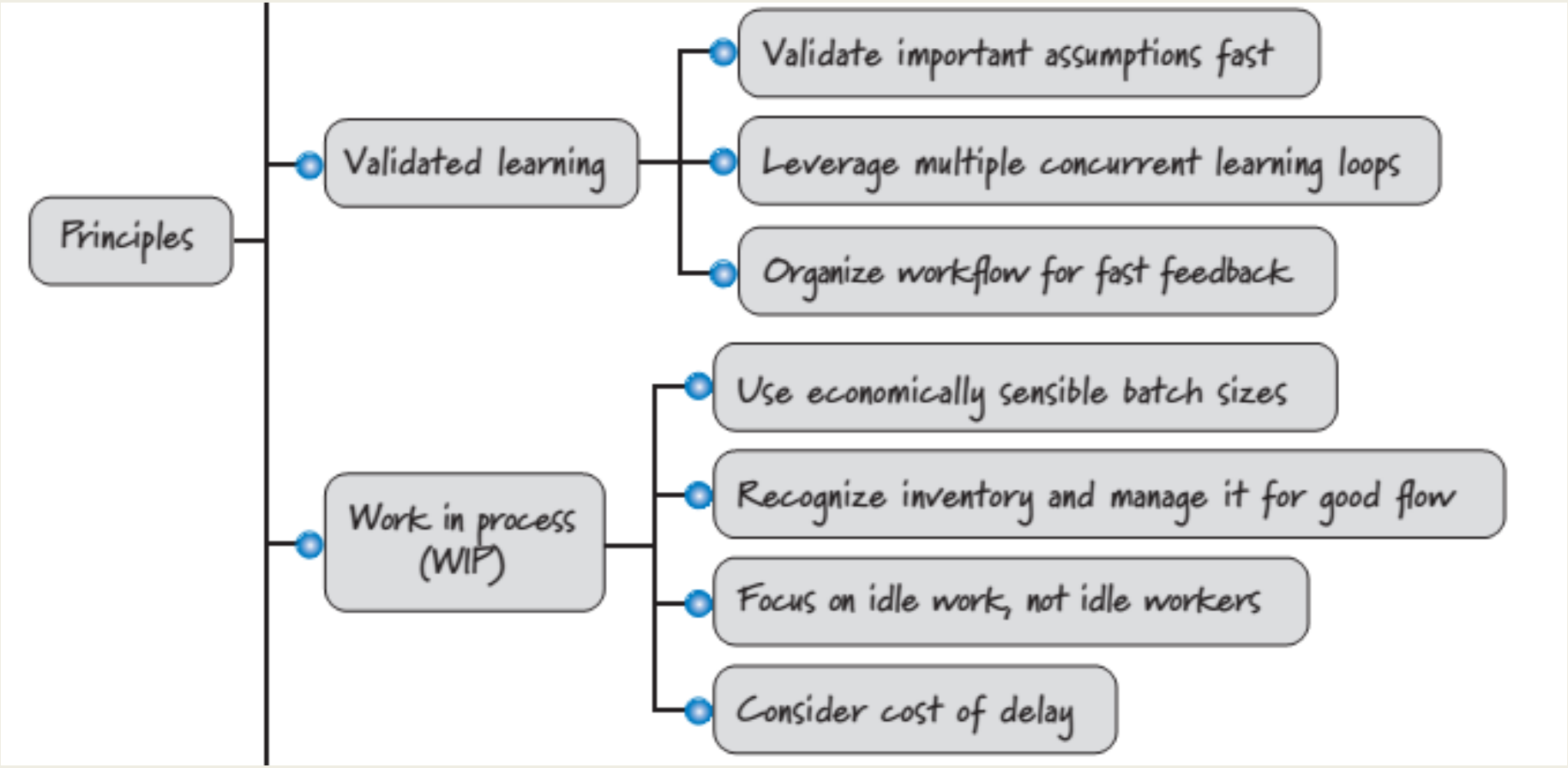
Isfahan University of Technology

mahmoudzadeh@iut.ac.ir

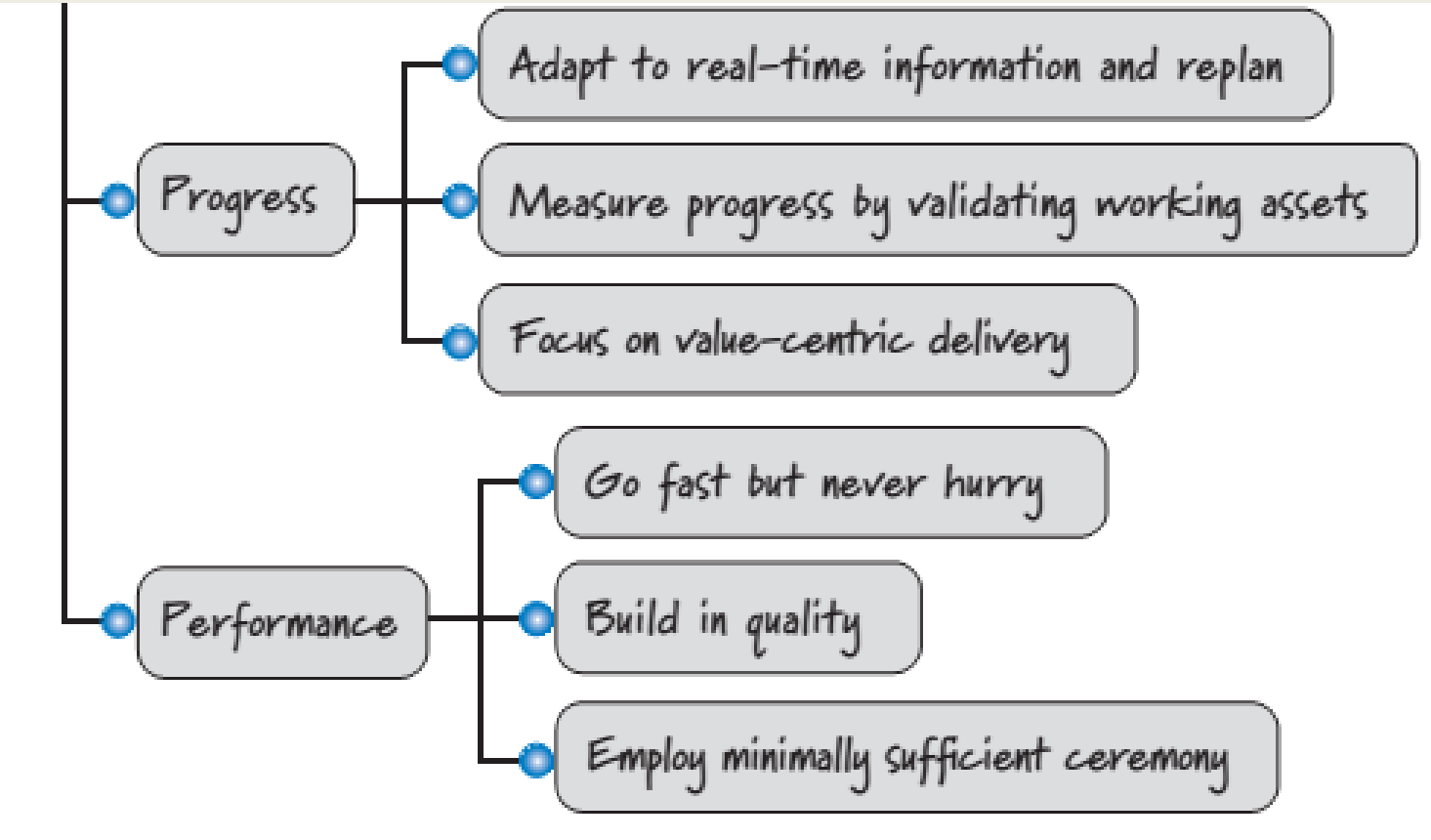
2020

Categorization of principles (Up)

Categorization of principles (Middle)



Categorization of principles (Bottom)



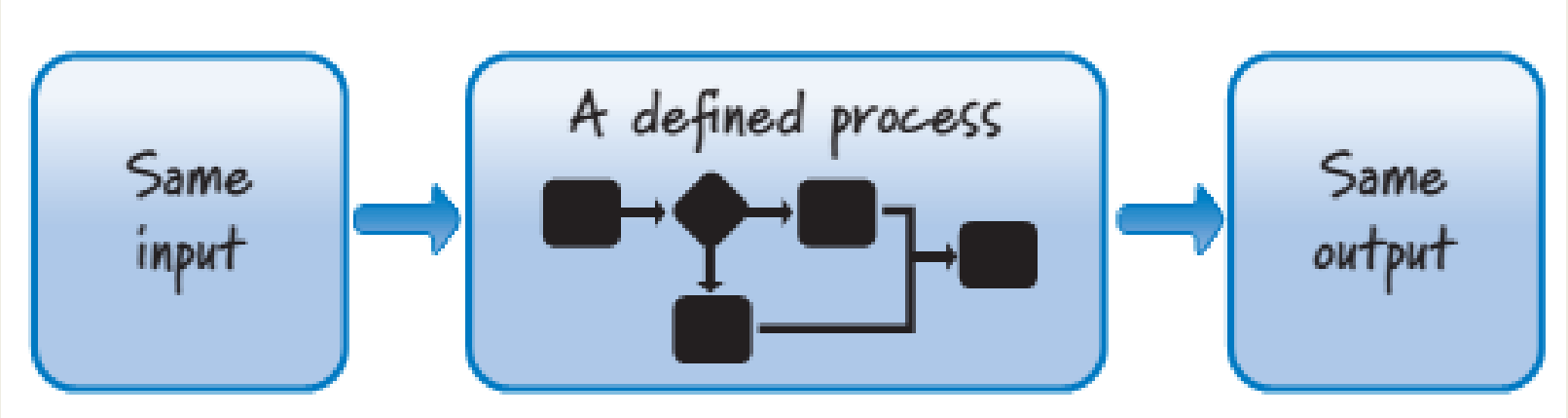
Variability and Uncertainty

- To create innovative solutions.
- Four related principles
  1. *Embrace helpful variability.*
  2. *Employ iterative and incremental development.*
  3. *Leverage variability through inspection, adaptation, and transparency.*
  4. *Reduce all forms of uncertainty simultaneously.*

-
تغییر پذیری و عدم قطعیت رو به عنوان یک اصل بپذیریم که خودش 4 تا زیراصل داره:

Embrace Helpful Variability (I)

- Plan-driven processes treat product development like manufacturing. In manufacturing our goal is to take a fixed set of requirements and follow a sequential set of well-understood steps to manufacture a finished product that is the same (within a defined variance range) every time.
- In product development, the goal is to create the unique single instance of the product, not to manufacture the product. This single instance is analogous to a unique recipe.



-

1- توی رویکردهای سنتی نوع نگاهی که به محصول نرم افزاری داشتن مثل اون نگاهی بود که به محصولات فیزیکی توی کارخانه ها داشتن

توی manufacturing یا تولید ما فیکس 3 تا استپ داریم ینی یه سری استپ هایی هستن که به صورت فیکس و ثابت به صورت متوالی باید انجام بشن--> ینی ما یک input داریم و یک فرایندی است و یک output میده اگر همین input رو بدیم به همون فرایند یک output دیگه به ما میده

نکته: توی محصولات فیزیکی ینی محصولاتی که توی کارخانه ها هستن این Variability کاملا داریم حذفش میکنیم ولی توی تولید محصولات نرم افزاری اینطوری نیست ینی قرار نیست محصولمون کپی باشه از محصولات قبلی --> هدف اینه که منحصر به فرد باشه اون محصول

Embrace Helpful Variability (II)

- We don’t want to create the same recipe twice;
- Instead, we want to create a unique recipe for a new product. Some amount of variability is necessary to produce a different product each time.
- In fact, every feature we build within a product is different from every other feature within that product, so we need variability even at this level.

-
ما نمی خوایم دقیقا عین همون محصول تولید بشه بلکه میخوایم یک محصول جدیدی که یک تغییری نسبت به قبلی داره رو انجام بدیم و بسازیم
فیچرهایی که در حقیقت هستن و نوع سرویسی که به end user میدیم و نحوه طراحی یوزر اینترفیس و حتی سرویس های دیگه که به سیستم اضافه میکنیم اخرش یه میزان تغییرپذیری رو ما داریم و نمی تونیم کپی پیست از اون محصول قبلی باشیم

Employ Iterative and Incremental Development (I)

- Plan-driven, sequential development assumes that we will get things right up front and that most or all of the product pieces will come together late in the effort.

- Scrum, on the other hand, is based on iterative and incremental development.

-
این هم از مواردی است که مدیریت میکنه اون Variability رو
توی متوالی یا Plan-driven این دور رو در نظر نمی گیره و می گه از اول کار باید همه چیز رو بدونیم و بعد توی مراحلی که می ریم جلو باید به همون چیز بپردازیم و کاملش بکنیم

از سوی دیگر، اسکرام مبتنی بر توسعه تکراری و افزایشی است.

Iterative Development (I)

- Acknowledges that we will probably get things wrong before we get them right and that we will do things poorly before we do them well (Goldberg and Rubin 1995).
- Is a planned rework strategy. We use multiple passes to improve what we are building so that we can converge on a good solution.
- Iterative development is an excellent way to improve the product as it is being developed.
- The biggest downside is that in the presence of uncertainty it can be difficult up front to determine (plan) how many improvement passes will be necessary.

--

* Iterative and Incremental

روش مدرن تولید با این Iterative and Incremental باشه ← اصول objectoried باشه

به ما این اجازه رو می ده که system رو step به step پیش پیش جلو

* Iterative Development

معمول ترین ضعف اسم اینه که Iterative معمولا زیاد باشن ← پیاده سازی

prototype است یعنی یک مرزمی تند سریع داریم و داریم اینو هی بررسی می کنیم

* Incremental Development

یک بخشی رو کامل انجام بده و بعد برو جلو ← یعنی بجای اینکه روی کل کار انجام

بده روی یک بخش انجام می ده و باز مورد می گیره و قسمت های بعدی رو تکمیل می کنه

از اون بخش

معمول ترین ضعف اسم ← کوچکی داریم به system نگاه می کنیم یعنی داریم شاخ و برگ

می بینیم بجای کل درخت

Iterative Development:

- یک استراتژی مجدد برنامه ریزی شده است.
- ما از چندین پاس برای بهبود چیزی که میسازیم استفاده میکنیم تا بتوانیم روی یک راهحل خوب همگرا شویم

Iterative and Incremental:

این بدان معناست که تحلیلگران سیستم درک خود را از مشکل کاربر با ایجاد سه نمای معماری کم کم توسعه می دهند

Iterative Development (II)

- For example, we might start by creating a prototype to acquire important knowledge about a poorly known piece of the product. Then we might create a revised version that is somewhat better, which might in turn be followed by a pretty good version.

- In the course of writing this book, for example, I wrote and rewrote each of the chapters several times as I received feedback and as my understanding of how I wanted to communicate a topic improved.

برای مثال، در طول نگارش این کتاب، هر یک از فصلها را چندین بار نوشتم و بازنویسی کردم که بازخورد دریافت میکردم و درکم از اینکه چگونه میخواهم با یک موضوع ارتباط برقرار کنم بهتر شد.

Incremental Development (I)

- Based on the age-old principle of “Build some of it before you build all of it.” We avoid having one large, big-bang-style at the end of development.
- Instead, we break the product into smaller pieces so that we can build some of it, learn how each piece is to survive in the environment in which it must exist, adapt based on what we learn, and then build more of it.
- Incremental development slices the system functionality into increments (portions).
- Incremental development gives us important information that allows us to adapt our development effort and to change how we proceed.
- The biggest drawback to incremental development is that by building in pieces, we risk missing the big picture (we see the trees but not the forest).

Incremental Development

- For example, while writing this book, I wrote a chapter at a time and sent each chapter out for review as it was completed, rather than trying to receive feedback on the entire book at once.
- This gave me the opportunity to incorporate that feedback into future chapters, adjusting my tone, style, or delivery as needed.
- It also gave me the opportunity to learn incrementally and apply what I learned from earlier chapters to later chapters.

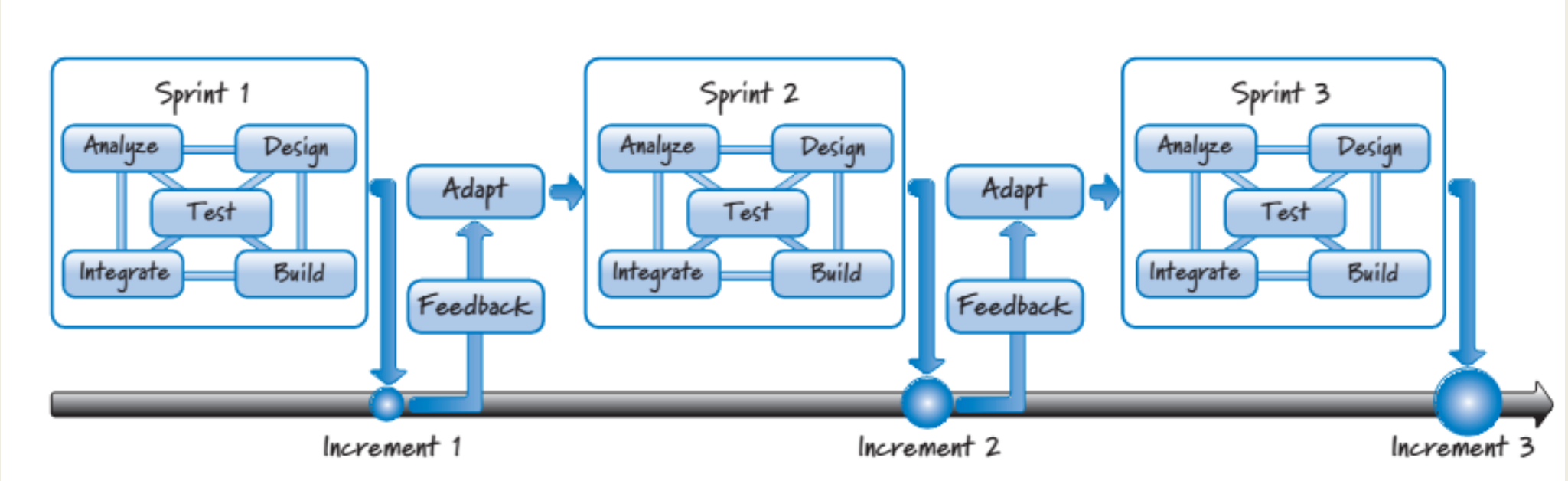
Iterative and Incremental Development in Scrum (I)

- Scrum leverages the benefits of both iterative and incremental development, while negating the disadvantages of using them individually.

- Scrum does this by using both ideas in an adaptive series of time boxed iterations called sprints.

-
توی اسکرام هر دو رو کنار هم در نظر می گیره ینی میگه که هم باید تکراری بریم جلو و هم افزایشی بریم جلو
پس توی اسکرام باید هر دو رو داشته باشیم که بتونیم به اون Variability and Uncertainty غلبه بکنیم و مدیریتش بکنیم وگرنه اگر از اول کار فیکسش بکنیم نمی تونیم عدم قطعیت رو مدیریت بکنیم و بهش بپردازیم

Iterative and Incremental Development in Scrum (II)



-
اینو در قالب اسلات های زمانی داره انجام میده:
یک اسلات زمانی است که اسمش رو sprint می ذاریم ینی ما یکسری کارهارو انجام میدیم برای همون Increment یک --> ینی خروجی sprintمون Incremental 1 است پس داریم Incremental می ریم جلو ینی اسلایس به اسلایس داریم کارو می بریم جلو در عین حال برمبنای بازخوردی که از اون Increment گرفتیم adapt میکنیم کار رو ینی Iterative هم می ریم جلو ینی اون ها رو تحویل میدیم، بازخورد میگیریم و برمبنای بازخورد کار رو اصلاح میکنیم ینی مرحله به مرحله داریم کار رو تکمیل و تکمیل ترش میکنیم ینی هم Iterative and Incremental است این ینی اگر تغییری کرد ما می تونیم توی راندهای بعدی اون تغییرات رو اعمال کنیم و اگر اطلاعاتی رو اول کار نداشتیم ینی اول کار یه سری چیزایی مبهم است ولی وقتی که بازخورد می گیریم ابهامات رفع میشه ینی هرچی داریم می ریم جلو Uncertainty به صفر میل میکنه
اسکرام زیرساختش طوری است که در قالب Iterative and Incremental به ما اجازه مدیریت Variability and Uncertainty به ما میده ینی می تونیم توی sprint های مختلفی هم Variability رو هندل بکنیم و هم Uncertainty پس هر دو رو باید با هم در نظر بگیریم

Iterative and Incremental Development in Scrum (III)

- During each sprint we perform all of the activities necessary to create a working product increment (some of the product, not all of it).
- This all-at-once approach has the benefit of quickly validating the assumptions that are made when developing product features.
- For example, we make some design decisions, create some code based on those decisions, and then test the design and code—all in the same sprint. By doing all of the related work within one sprint, we are able to quickly rework features, thus achieving the benefits of iterative development.

Iterative and Incremental Development in Scrum (IV)

- In Scrum, we don't work on a phase at a time; we work on a **feature** at a time.
- So, by the end of a sprint we have created a valuable product increment (some but not all of the product features). That increment includes or is integrated and tested with any previously developed features; otherwise, it is not considered done.
- At the end of the sprint, we can get feedback on the newly completed features within the context of already completed features. This helps us view the product from more of a big-picture perspective than we might otherwise have.

-
توی اسکرام روی فیچر کار میکنیم و روی فاز کار نمی کنیم
تمام کارهایی که برای رسیدن به اون فیچر انجام میدیم پس Incremental میریم جلو و در اخر sprint فیدبک می گیریم واین فیدبک باعث میشه که اگر فیچرها ایراد داشته باشن، ایراد رو در نظر بگیریمش پس هم داریم Iterative میریم جلو و هم Incremental

Iterative and Incremental Development in Scrum (V)

- We receive feedback on the sprint results, which allows us to adapt.
- We can choose different features to work on in the next sprint or alter the process we will use to build the next set of features.
- In some cases, we might learn that the increment, though it technically fits the bill, isn't as good as it could be. When that happens, we can schedule rework for a future sprint as part of our commitment to iterative development and continuous improvement.
- This helps overcome the issue of not knowing up front exactly how many improvement passes we will need.
- Scrum does not require that we predetermine a set number of iterations.
- The continuous stream of feedback will guide us to do the appropriate and economically sensible number of iterations while developing the product incrementally.

-
...
چون داریم مرحله به مرحله می ریم جلو نمی خواد اول کار ریزه کاری ها رو بدونیم و مرحله به مرحله بهش می پردازیم
برمبنای اون فیدبک هایی که میگیریم هم کار رو ارتقا میدیم و هم پیش بینی بکنیم با چند تا Iterative and Incremental کار رو داریم می بریم جلو

Leverage Variability through Inspection, Adaptation, and Transparency

- A plan-driven, sequential development process assumes little or no output variability. It follows a well-defined set of steps and uses only small amounts of feedback late in the process.
- Scrum embraces the fact that in product development, some level of variability is required in order to build something new.
- Scrum also assumes that the process necessary to create the product is complex and therefore would defy a complete up-front definition. Furthermore, it generates early and frequent feedback **to ensure that the right product is built and that the product is built right.**

-
Inspection: خروجی محصول و خروجی عملکرد خودمون رو بررسی میکنیم
Adaptation: برمبنای نتایج Inspection تطبیق میدیم ینی خودمون رو تنظیم بکنیم با اون چیزی که کارفرما می خواد - ممکنه فرایند رو درست نباشه و اون هم درستش میکنیم
Transparency: چیزی که خیلی خیلی مهم است توی انگیزه های ادم ها توی تیم حس اعتمادی است که ادم ها نسبت به هم دارند --> در راستای افزایش این اعتماده چیزی که خیلی مهم است شفافیت است
توی plan-driven ها Variability رو نداریم ینی اول کار یکسری استپ های از پیش تعریف شده ای داریم و می ریم تا اخر کار و اگر بخوایم فیدبکی بگیریم فیدبک هایی که توی اون مراحل میگیریم ینی چک پوینت هایی که اخر میگیریم چک پوینت های معمولا مفیدی نیست و اگر هم باشه میزان Variability اش خیلی کمه چون ما working software ارائه نمیدیم بلکه داریم داکیومنت ارائه میدیم
توی اسکرام Variability رو میاد برمبنای Inspection و Adaptation گارانتی میکنه
چجوری؟ میاد کار رو بررسی میکنه و برمبنای اون خروجی اون بررسی ها adapt میکنه ینی ما Inspection , Adaptation رو نیاز داریم

Comparison

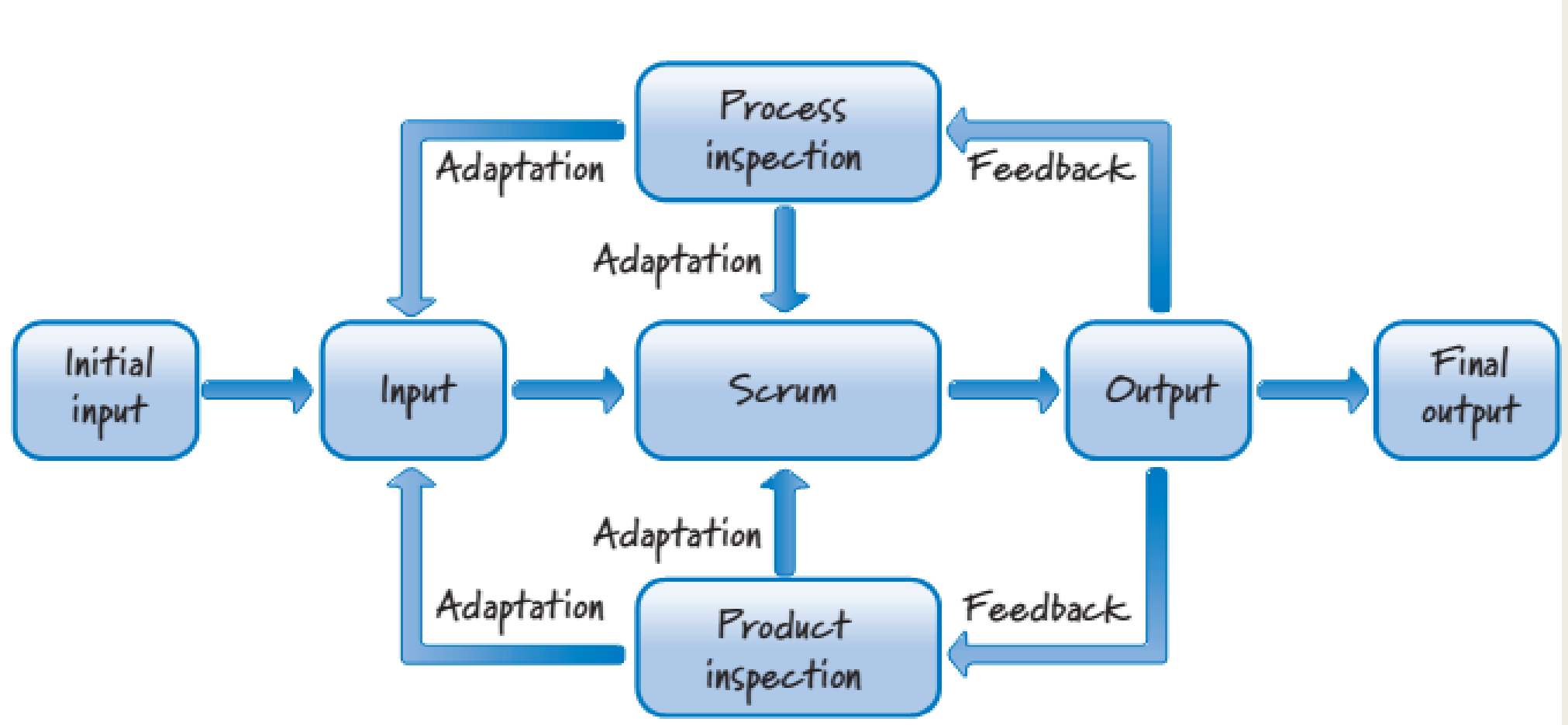
Dimension	Plan-Driven	Scrum
Degree of process definition	Well-defined set of sequential steps	Complex process that would defy a complete up-front definition
Randomness of output	Little or no output variability	Expect variability because we are not trying to build the same thing over and over
Amount of feedback used	Little and late	Frequent and early

مقایسه:
میزان یا درجه تعریف فرایند توی plan-driven را کاملا مشخص است ینی نزدیک به 100 درصد ینی اول کار باید همه چیز مشخص و واضح باشد ینی علاوه بر ریکوارمنت ها ما هم باید دقیق بگیم که برناممون چیه و روزی چند ساعت باید روی این مسئله وقت بذاریم
ولی توی اسکرام نه میگه فرایند یک فرایند پیچیده ای است و از اول نمی تونیم که همه چیز رو داشته باشیم

randomness of output : میزان تصادفی بودن output ینی میزان فیکس نبودن اون خروجی اون فرایند --> توی plan-driven ها یا صفر است یا کم است ولی توی اسکرام نه ینی ما به ما این اجازه رو میده که هرباری که بازخورد می گیریم بر مبنای اون بازخورد تنظیم بکنیم کار رو

میزان فیدبک: توی plan-driven ها خیلی کم و حتی جزئی است ولی توی اسکرام frequently یا مکررا داریم بازخورد رو دریافت میکنیم

Scrum Process model



اینجا یک initial input داریم که یک مقداری روش کار میکنیم و بعد پروپوزال رو تهیه میکنیم بعد می رسیم input وبعد به اسکرام می رسیم و بعد می رسیم به output
این output همون Increment است ینی اخر هر اسکرام یک Increment داریم این Incremen ما هم بازخورد روی محصول می گیریم و بعد برمبنای این بازخورد هم می تونه روی اسکرام اثر بذاره ینی محصول ممکنه یکی دوتا تسکش کم یا زیاد بشه و بعد لیست کارهایی که اینجا باید انجام بشه تغییر بکنه
یک طرفش دیگش اینه که اون بازخورد اون پروسس رو داریم ینی ممکنه به این نتیجه برسیم که رسیدیم به اون تعهدی که داشتیم ینی مثلا 90 درصد کار رفته جلو و تست هاش هنوز انجام نشده بعد اینو Inspect میکنیم پروسس رو ینی تیم مشکلی نداره و ادم هایی توی تیم مشکلی ندارن و .. یا ممکنه همه خوب کار کردن ولی نظر تیم این بوده که 80 ساعت وقت می خواسته ولی در واقعیت 100 ساعت وقت می خواسته

برای اینکه بتونیم پذیرای تغییرات باشیم مجبوریم این بازبینی و تغییر رو توی اسلات های کوچیک انجام بدیم
برمبنای این اصول اسکرام تنظیم شده ینی برمبنای اینکه بتونه تغییرات رو پاسخ بده چرخه های کوچیک اسکرام اومده طراحی شده

Inspect, Adapt and Transparency

- Are heart of Scrum.

- In Scrum, we inspect and adapt not only what we are building but also how we are building it

بازرسی، تطبیق و شفافیت
قلب اسکرام هستند.

در اسکرام، ما نه تنها آنچه را که میسازیم، بلکه نحوه ساخت آن را نیز بازرسی و تطبیق میدهیم

Transparency

- All of the information that is important to producing a product must be available to the people involved in creating the product.
- Transparency makes inspection possible, which is needed for adaptation.
- Transparency also allows everyone concerned to observe and understand what is happening.
- It leads to more communication and it establishes trust (both in the process and among team members).

شفافیت

+ تمام اطلاعاتی که برای تولید یک محصول مهم است باید در دسترس افرادی باشد که در ایجاد محصول نقش دارند. --> اطلاعات باید به یک میزان در دسترس همه باشه و کسی بردیگری برتری نداره

+ شفافیت بازرسی را ممکن می کند، که برای انطباق لازم است.

+ شفافیت همچنین به همه افراد مرتبط اجازه می دهد تا آنچه را که اتفاق می افتد مشاهده و درک کنند.

+ منجر به ارتباطات بیشتر و ایجاد اعتماد (هم در فرآیند و هم در بین اعضای تیم) می شود.

Reduce All Forms of Uncertainty Simultaneously (I)

- Developing new products is a complex endeavor with a high degree of uncertainty.
- Three categories of uncertainty.
  1. *End uncertainty (what uncertainty)—uncertainty surrounding the features of the final product.*
  2. *Means uncertainty (how uncertainty)—uncertainty surrounding the process and technologies used to develop a product.*
  3. *Customer uncertainty (who uncertainty) —who the actual customers of their products will be.*
     - This uncertainty must be addressed or they might build brilliant products for the wrong markets.

-
توی اسکرام همه انواع عدم قطعیت رو همزمان مدیریت میکنه یا کاهش میده
انواع عدم قطعیت:
1- خود محصول: ممکنه خود کارفرما ندونه - ممکنه ادبیات ها یکسان نباشه --> که بهش میگیم what uncertainty ینی ان چیزی که باید بهش برسیم رو نمیدونیم چیه
2- دومیش مربوط به how uncertainty است ینی روند انجام کار--> زمان رو نمیدونیم و هزینه هاش رو نمیدونیم و ممکنه خطا داشته باشه و ممکنه ارتباط بین ادم ها تغییر بکنه
3- سومیش مربوط به who uncertainty است ینی مشتری ها هم می تونن تغییر بکنن --> ینی ممکنه جامعه هدف و بازار تغییر بکنه
بهترین محصولات ممکنه برای بازار اشتباهی طراحی و پیاده سازی بشه
با این سه تا چجوری بریم جلو؟

Reduce All Forms of Uncertainty Simultaneously (II)

- Traditional, sequential development processes focus first on eliminating all end uncertainty by fully defining up front what is to be built, and only then addressing means uncertainty.
- This simplistic, linear approach to uncertainty reduction is ill suited to the complex domain of product development, where our actions and the environment in which we operate mutually constrain one another.

-
رویکردهای سنتی میان عدم قطعیت رو حذف می کنن ینی فرض میکنن که محصول رو از اول می دونن چیه و همه ریکوارمنت های سیستم رو کامل در میارن و اینطوری اومدن اینکه محصول چه باید باشد رو مدیریت کردن ینی می دونیم محصول چیه و بعد تمام تمرکز رو می ذاریم روی بقیه
--> پس با محدود کردن یا حذف یکی از عدم قطعیت ها روی دیگری ها متمرکز میشیم
این کار منطقا برای پروژه هایی که عدم قطعیتش بالاست اشتباه است چون ما داریم محصولی تولید میکنیم که این یک اکشنی انجام میده و این اکشن رو به کارفرما و محیط ارائه میدیم و محیط به ما بازخورد میده ینی فیچر یا اکشنی رو که انجام دادیم رو برای اون اکشن یک بازخورد می گیریم و برمبنای اون بازخورد میایم اکشن رو عوض میکنیم و این اکشن عوض شده رو دوباره به محیط ارائه میدیم این ینی اینکه فیچرما و محیط ما در تعامل با هم دیگه هستن و نمیتونیم یکی رو فیکس کنیم و به اون یکی بپردازیم و وقتی به یکی می پردازیم درگیر ددومی هم میشیم پس نمیتونیم این اتفاق رو هندلش بکنیم
محیط --> در واقع همه مخاطبین ما میشن که دارن به ما بازخورد میگن

پس منطق plan-driven ها توی مدیریت عدم قطعیت اشتباه است

Reduce All Forms of Uncertainty Simultaneously (III), An Example

- We decide to build a feature (our action).
- We then show that feature to a customer, who, once he sees it, changes his mind about what he really wants, or realizes that he did not adequately convey the details of the feature (our action elicits a response from the environment).
- We make design changes based on the feedback (the environment's reaction influences us to take another unforeseen action).

Reduce All Forms of Uncertainty Simultaneously (IV)

- In Scrum, we do not constrain ourselves by fully addressing one type of uncertainty before we address the next type.
- Instead, we focus on simultaneously reducing all uncertainties (end, means, customer, and so on).
- Of course, at any point in time we might focus more on one type of uncertainty than another. Simultaneously addressing multiple types of uncertainty is facilitated by iterative and incremental development and guided by constant inspection, adaptation, and transparency.
- Allows to opportunistically probe and explore our environment to identify and learn about the unknown unknowns(the things that we don't yet know that we don't know) as they emerge.

-
توی اسکرام میاد برمبنای همین منطق همه عدم قطعیت ها رو با هم مدیریت میکنه ینی توی هر اسلات زمانی هم به what , how , who می پردازه ینی همزمان با هم این کارو انجام میده و ما محدود نمیکنیم خودمون رو که به یک نوعی از عدم قطعیت رو فیکس بکنیم و به بقیه بپردازیم و میایم به صورت همزمان این کارو انجام میدیم
چجوری ؟ توی هر اسلات زمانی به صورت تکراری و تدریجی این کارو انجام میده ینی میاد فیچر رو ارائه میده و بازخورد رو از محیط میگیره و ممکنه این بازخوردی که میگیره نیاز به این باشه که خود فیچر عوض بشه یا نیاز به این باشه که how عوض بشه یا ممکنه بازار عوض بشه پس برمبنای iterative and incremental میتونیم به طور همزمان این سه نوع عدم قطعیت رو مدیریت کنیم که بیس این ها هم Adaptation و Inspection است

Reference

1- K. S. Rubin, “Essential Scrum, A Practical guide to the most popular agile process,” 2013.