بسمه تعالی

فصل پنجم

فرآیند تولید نرم‌افزار در متدولوژی USDP

اهداف جلسه

- آشنائی با روش‌های مدرن توسعه نرم‌افزار
- معرفی فرآیند تولید در متدولوژی USDP
- درک تفاوت بین USDP و RUP
- معرفی محورهای اصلی USDP
- درک محصول بودن RUP
- آشنائی با ابعاد RUP

فهرست مطالب

- فرآیند تولید نرم‌افزار
- روش‌های مدرن توسعه نرم‌افزار
- روش‌های سنگین و سبک
- مقدمه‌ای بر فرآیند USDP
- محورهای اصلی USDP
- RUP بعنوان یک محصول
- معرفی ابعاد فرآیند RUP

فرآیند تولید نرم‌افزار

- یک فرآیند تولید نرم‌افزار چهار نقش اساسی دارد
  - مشخص نمودن ترتیب فعالیت‌هایی که باید صورت گیرد تا نیازمندی‌های کاربران به یک محصول واقعی تبدیل شوند
  - بیان اینکه چه فرآورده‌هایی باید تولید شوند و در چه زمانی
  - تعیین روش اداره وظایف توسعه‌دهندگان منفرد و تیمی، نقش‌های مورد نیاز در پروژه و انتساب این نقش‌ها به اعضای تیم
  - فراهم نمودن معیارهایی (*Software Metrics*) برای اندازه‌گیری کیفیت محصولات پروژه و روند پیشرفت فعالیت‌های آن

فرآیند تولید نرم‌افزار (ادامه)

در فرآیند تولید که

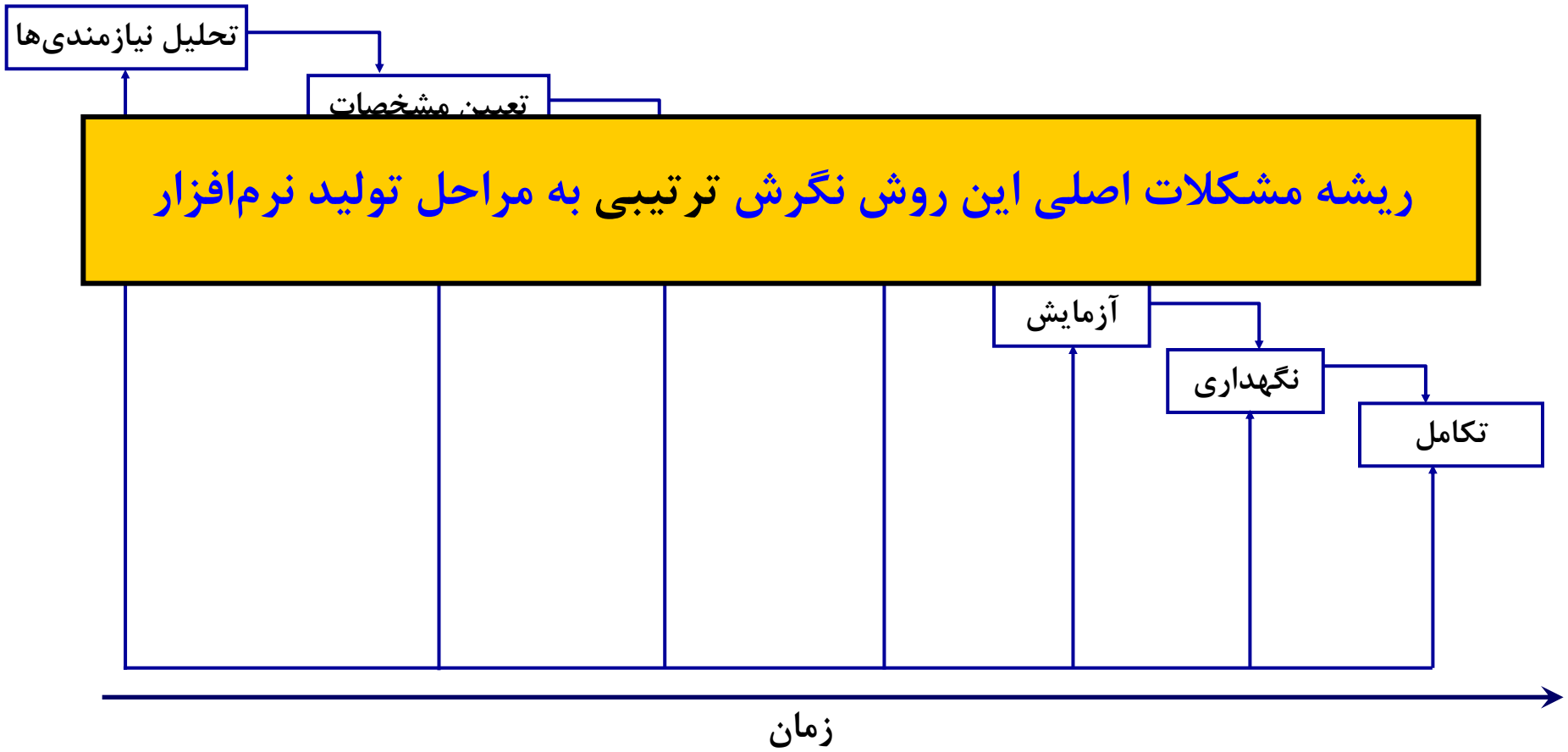
- به خوبی مستند شده باشد ← نرم‌افزارهای مورد نیاز به صورت منظم و قابل پیش‌بینی قابل تولیدند
- به خوبی مستند نشده باشد ← موفقیت بستگی به تلاش طاقت فرسای اعضای تیم دارد

فرآیند تولید نرم‌افزار خوب

- یکی از مشخصات بارز یک فرآیند تولید خوب استفاده از تجربیات بدست آمده از اجرای پروژه‌های نرم‌افزاری موفق است

تکرار و توسعه تدریجی

فرآیند آبشاری (*Waterfall Process*)



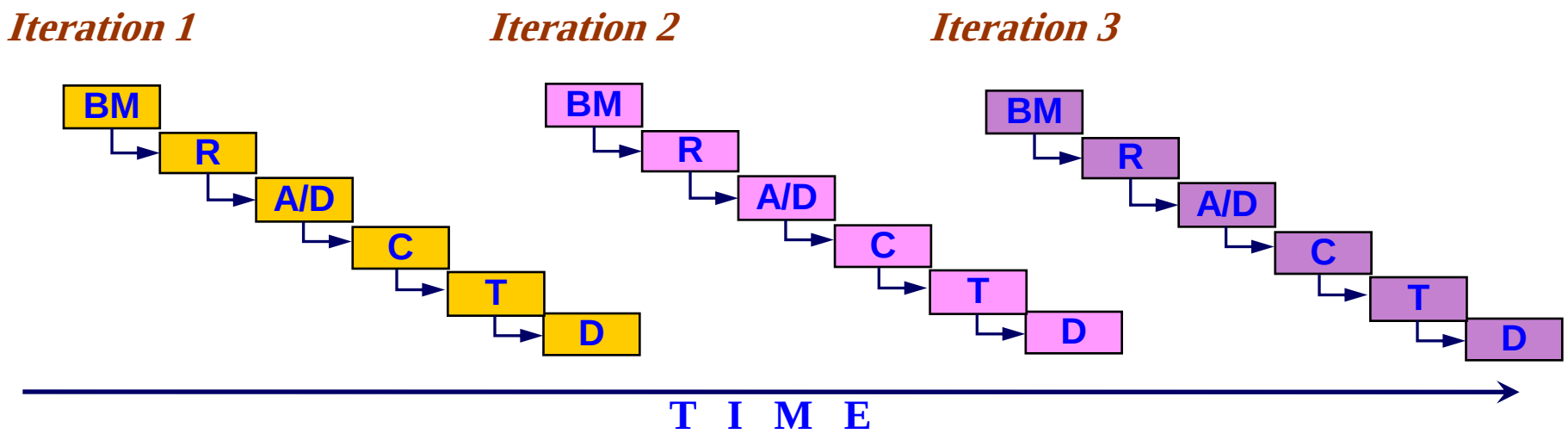
تکرار و توسعه تدریجی (ادامه)

هزینه ریسک در روش آبشاری

تکرار و توسعه تدریجی (ادامه)

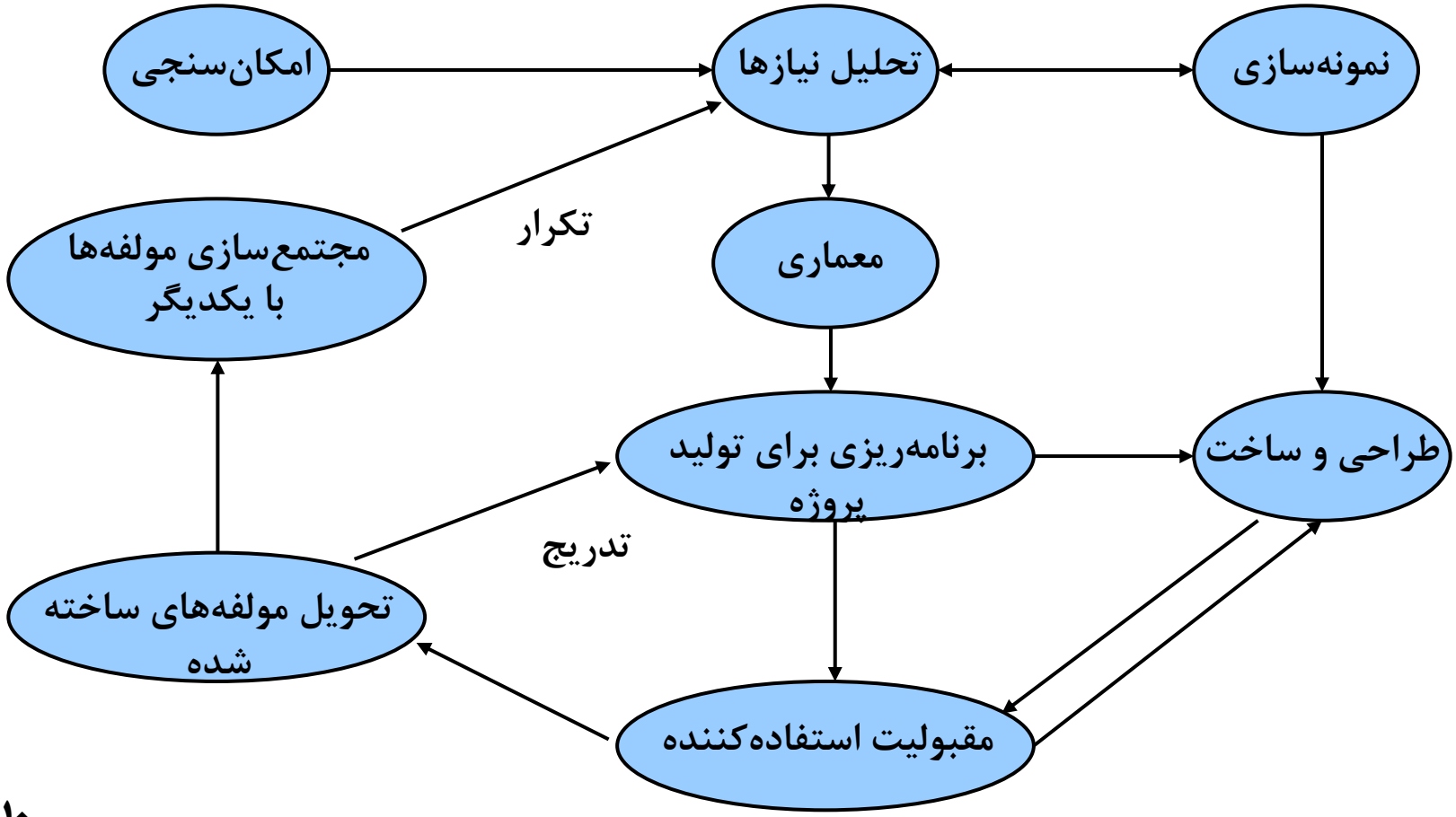
BM: Business Modeling
R: Requirements Analysis
A/D: Analysis & Design
C: Implementation
T: Test
D: Deployment

روش تکرار و توسعه تدریجی



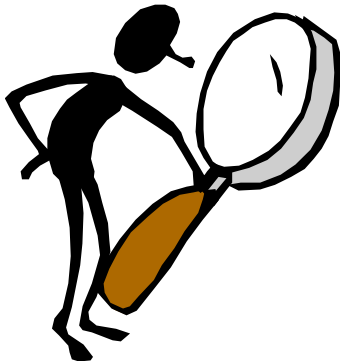
تکرار و توسعه تدریجی (ادامه)

روش تکرار و توسعه تدریجی



تکرار و توسعه تدریجی – ویژگیها

- تشخیص زود هنگام خطاهایی که در درک مسأله، تحلیل یا طراحی رخ می‌دهند
- تشخیص زودهنگام ناسازگاری‌های موجود بین تحلیل نیازمندی‌ها، طراحی و پیاده‌سازی
- کاربر می‌تواند دائما از روند پیشرفت پروژه مطلع گردد

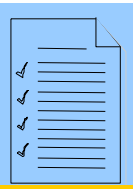


تکرار و توسعه تدریجی – ویژگیها (ادامه)

- بوسیله این روش می‌توان روی قسمت‌های مهمتر پروژه متمرکز شد و از پرداختن به قسمت‌های کم اهمیت‌تر پرهیز نمود
- آزمایش تکراری و مستمر امکان تشخیص بهتر روند پیشرفت پروژه را به ما می‌دهد
- بارکاری (*Workload*) تیم‌ها، بخصوص آزمایش‌کنندگان، روی چرخه تولید پروژه به صورت متوازن توزیع می‌شود

مدیریت نیازمندی‌ها

- نیازمندی‌ها عبارتست از شرطی یا قابلیتی که سیستم باید دارای آن باشد



یکی از ثابت‌ها در دنیای نرم‌افزار متغیر و پویا بودن نیازمندی‌هاست

۲) اعمال تغییرات مطلوب روی نیازمندی‌های جمع‌آوری شده

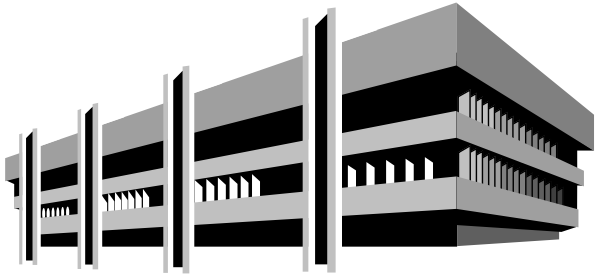
۳) ردیابی و مستندکردن اثرات بوجود آمده و تصمیم‌های اتخاذ شده

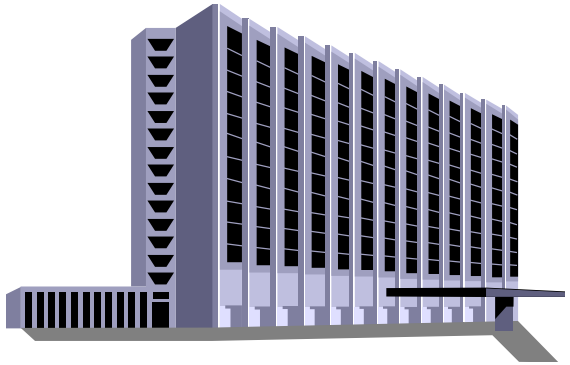
مدیریت نیازمندی‌ها

- ویژگیها
  - یک روش منظم و سیستماتیک برای مدیریت نیازمندی‌ها
  - در این روش می‌توان نیازمندی‌ها را اولویت‌بندی، فیلتربندی یا ردیابی نمود
  - امکان تشخیص واقعی و منصفانه عملکرد و کارایی سیستم وجود دارد
  - ناسازگاری‌ها به آسانی قابل کشفند

استفاده از معماری مبتنی بر مؤلفه‌ها

- معماری سیستم عبارتست از تعیین ساختار کلی سیستم و روش‌هایی که این ساختار را قادر به تامین کلیه ویژگیهـای کلیدی سیستم (*cross-cutting concerns*) می‌سازد





استفاده از معماری... (ادامه)

- معماری سیستم شامل تصمیم‌گیری‌هایی در سطح کلان در

علاوه بر ساختار و رفتار سیستم، معماری با مواردی از قبیل کارایی، انعطاف‌پذیری، استفاده مجدد و محدودیت‌های تکنولوژی و اقتصادی نیز سروکار دارد

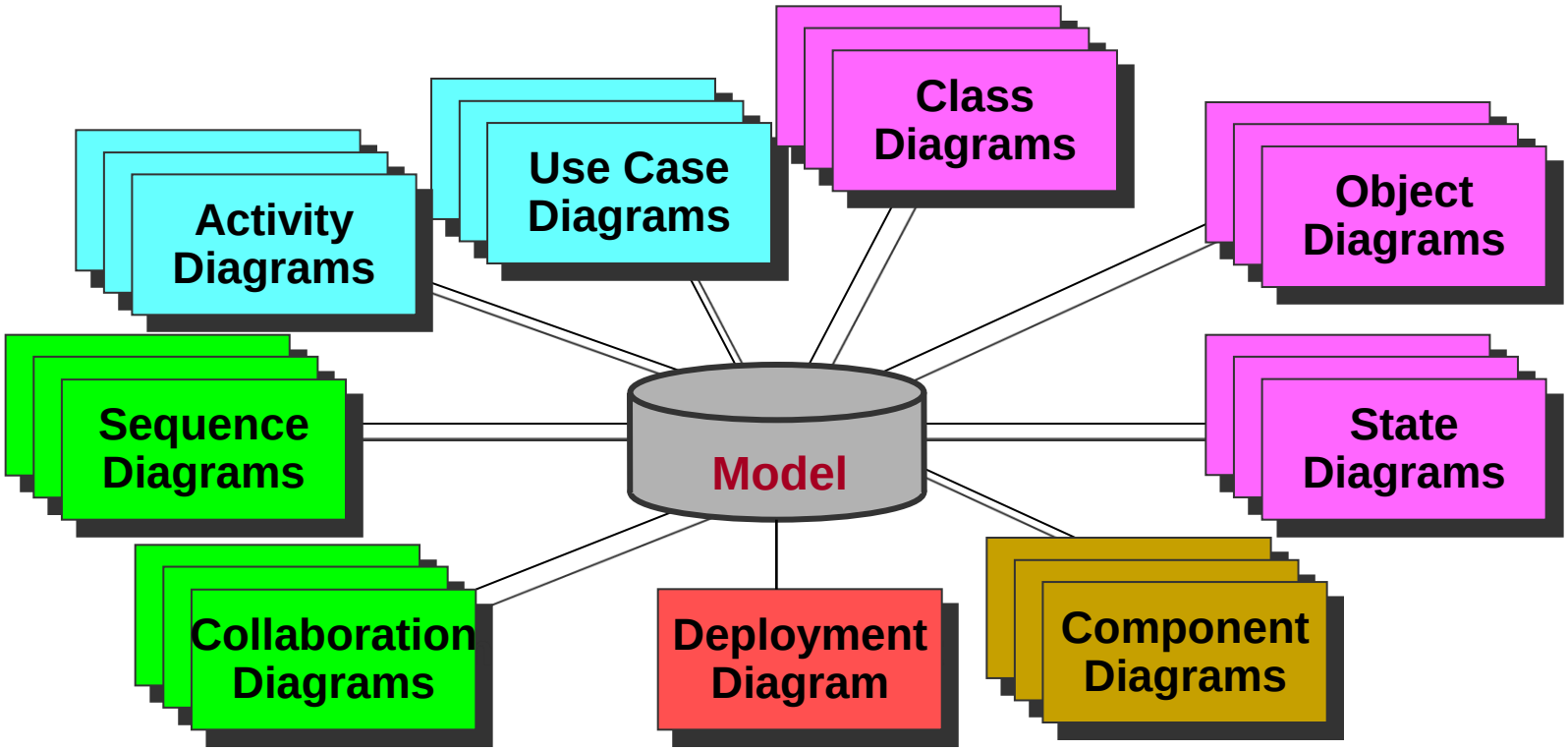
- سبک معماری مورد استفاده

استفاده از معماری... (ادامه)

- یکی از شیوه‌های مهم معماری نرم‌افزار، توسعه مبتنی بر مؤلفه‌ها است زیرا این روش امکان استفاده مجدد از آنها را به ما می‌دهد
- ویژگیها
  - کمک در داشتن یک معماری کشسان
  - افزایش قابلیت استفاده مجدد از مؤلفه‌ها و تکنولوژی‌های موجود
  - مؤلفه، پایه خوبی برای مدیریت پیکربندی است

مدلسازی تصویری نرم‌افزار

- مدل عبارت از یک توصیف ساده شده، با توجه به یک نگرش معین، از سیستم است

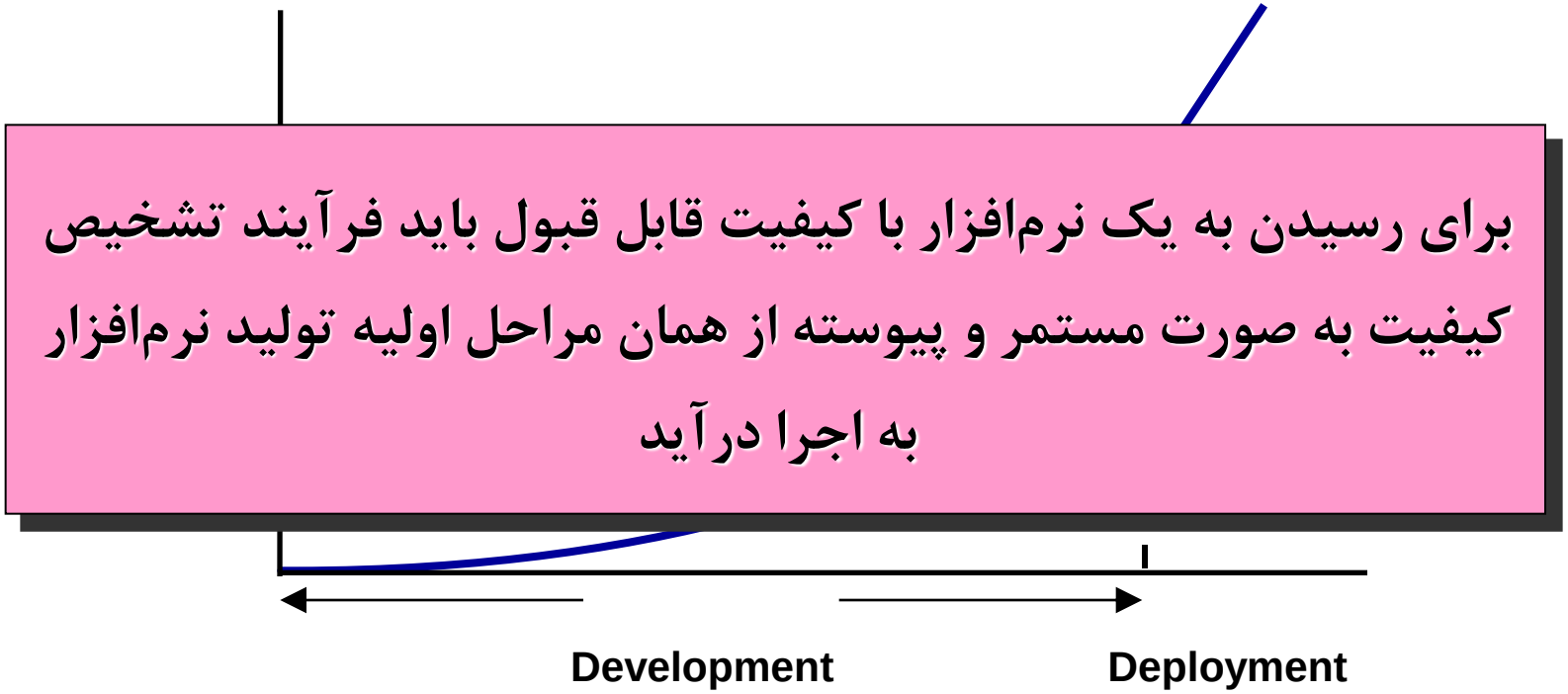


مدلسازی تصویری نرم‌افزار (ادامه)

- ویژگیها
  - امکان توصیف سیستم با میزان دلخواهی از جزئیات
  - بوسیله مدل‌ها می‌توان طراحی سیستم را به صورت روشن و صریح بیان نمود
  - امکان تشخیص معماری‌های غیر قابل انعطاف و واحدبندی نشده
  - نرم‌افزار خوب حاصل مدل‌های با کیفیت بالا

بررسی کیفیت نرم‌افزار

برای رسیدن به یک نرم‌افزار با کیفیت قابل قبول باید فرآیند تشخیص کیفیت به صورت مستمر و پیوسته از همان مراحل اولیه تولید نرم‌افزار به اجرا درآید



هزینه اصلاح خطاها به صورت نمایی رشد می‌نماید

بررسی کیفیت نرم‌افزار (ادامه)

- ویژگیها
  - فرآیند تشخیص روند پیشرفت پروژه مبتنی بر واقعیت‌ها، نه بر حدس‌ها و محاسبات کاغذی خواهد بود
  - این فرآیند ناسازگاری‌های موجود بین نیازمندی‌ها، طراحی و پیاده‌سازی را آشکار می‌سازد
  - امکان کشف زود هنگام خطاها را به ما می‌دهد و بدین صورت هزینه اصلاح آنها بشدت کاهش می‌یابد

مدیریت پیکربندی

- هنر تشخیص، سازماندهی و کنترل تغییراتی که برای نرم‌افزار در مدت کارکرد خود (از ابتدای تولید تا خارج شدن از عمل) رخ می‌دهند
- یکی از مشکلات اساسی تولید نرم‌افزار مدیریت پیکربندی است
  - این مشکل بویژه در پروژه‌های بزرگ که در آن تیم‌های متعددی با یکدیگر بر روی تکرارها، نشرها، محصولات و سکوهای متفاوت کار می‌کنند، قابل مشاهده است

مدیریت پیکربندی (ادامه)

- ویژگیها
  - یک روش سیستماتیک و قابل‌تکرار برای کنترل تغییرات نرم‌افزار
  - کنترل انتشار تغییرات
  - کاهش تداخل بین کار توسعه‌دهندگان که به صورت موازی با هم کار می‌کنند
  - نرخ تغییر معیار مناسبی برای تشخیص وضعیت فعلی پروژه است

متدولوژی‌های توسعه نرم افزار

- متدولوژی‌های سنگین وزن (*Heavyweight*)
  - این نوع متدولوژی‌ها معمولاً مستندات، محصولات و فرآورده‌های بیشتری تولید می‌نمایند
- متدولوژی‌های سبک وزن (*Lightweight*)
  - تاکید بر تولید محصول نهایی و نه فرآورده‌ها و مستندات بسیار است

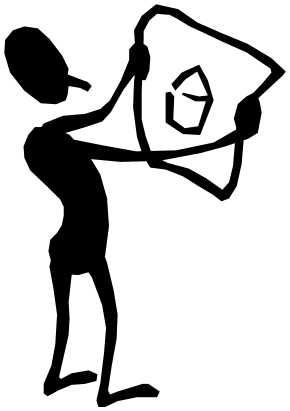
مقایسه متدولوژی‌های سبک و سنگین

- معیارهای مقایسه متدولوژی‌ها با یکدیگر
  - روش
  - معیار موفقیت
  - اندازه پروژه
  - سبک مدیریت
  - نحوه مستندسازی
  - چرخه‌ها
  - اندازه تیم
  - برگشت سرمایه

مقایسه متدولوژی‌ها – روش

- روش‌های سریع‌الانتقال بصورت *Adaptive* یا ســازگار عمــل
  می‌ک
- روش عمــل
  می‌ک بینــی
  کنند

آیا همه چیز از ابتدا قابل پیش‌بینی است؟



مقایسه متدولوژی‌ها – معیار موفقیت

- معیار موفقیت در روش‌های سریع‌الانتقال دستیابی به ارزش کاری
- در ر... تن در راست...

روش‌های سنگین وزن از نظر تغییر در فرآیند توسعه در حین پروژه انعطاف‌پذیری ندارند

مقایسه متدولوژی‌ها – اندازه

- اندازه پروژه در روش‌های سریع‌الانتقال کوچک است
- ا

این مسأله از محبوبیت روش‌های سریع‌الانتقال نمی‌کاهد

(آمار نشان می‌دهد که تعداد پروژه‌های کوچک بسیار بیشتر است)

مقایسه متدولوژی‌ها – سبک مدیریت

- مدیریت در روش‌های سریع‌الانتقال بصورت غیرمتمرکز و
- و

مدیریت غیرمتمرکز امکان تصمیم‌گیری بهتر را در حین پروژه فراهم می‌کند

مقایسه متدولوژی‌ها – مستندسازی

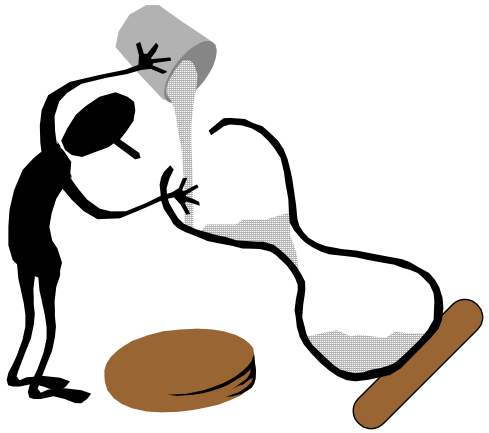
- مستندسازی در روش‌های سریع‌الانتقال بصورت بسیار

در بسیاری از موارد مستندسازی‌های سنگین, کار بسیار دشوار و زمانبری است

مقایسه متدولوژی‌ها – چرخه‌ها

- تعداد چرخه‌ها (*Cycles*) در روش‌های سریع‌الانتقال بســیار

زمانبر بودن چرخه‌های تولید, موجب طولانی شدن زمان انتظار برای رسیدن به نشرها می‌شود

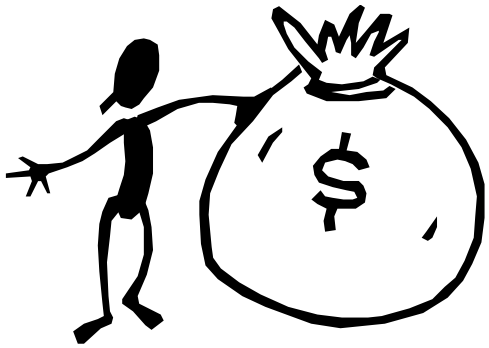


مقایسه متدولوژی‌ها – اندازه تیم

- در روش‌های سریع‌الانتقال اندازه تیم کوچک است (بین ۲۰
- خلاقیت و همکاری در تیم کوچک بسیار بیشتر خواهد بود

مقایسه متدولوژی‌ها – برگشت سرمایه

- در روش‌های سریع‌الانتقال سرمایه خیلی زود در طول پروژه
- روش‌های سریع‌الانتقال از لحاظ اقتصادی بصرفه‌اند



معرفی USDP

- یک فرآیند تولید مهندسی نرم‌افزار است که روشی منظم و سیستماتیک برای ترتیب انجام فعالیت‌ها در یک پروژه

فرآیند USDP از مدل شی‌گرایی حمایت نموده و پایه روش‌های مدرن توسعه نرم‌افزار است

- [illegible]

سنتی و معمولی گرفته تا سیستم‌های هوشمند و سیستم‌های اطلاعاتی بزرگ را دربردارد

محورهای اصلی USDP

- سه محور اصلی USDP
  - راهبری بر مبنای موارد کاربری (*Use-Case Driven*)
  - محوریت قرار دادن معماری (*Architecture Centric*)
  - استفاده از روش تکرار و توسعه تدریجی

    (*Iterative and Incremental Development*)

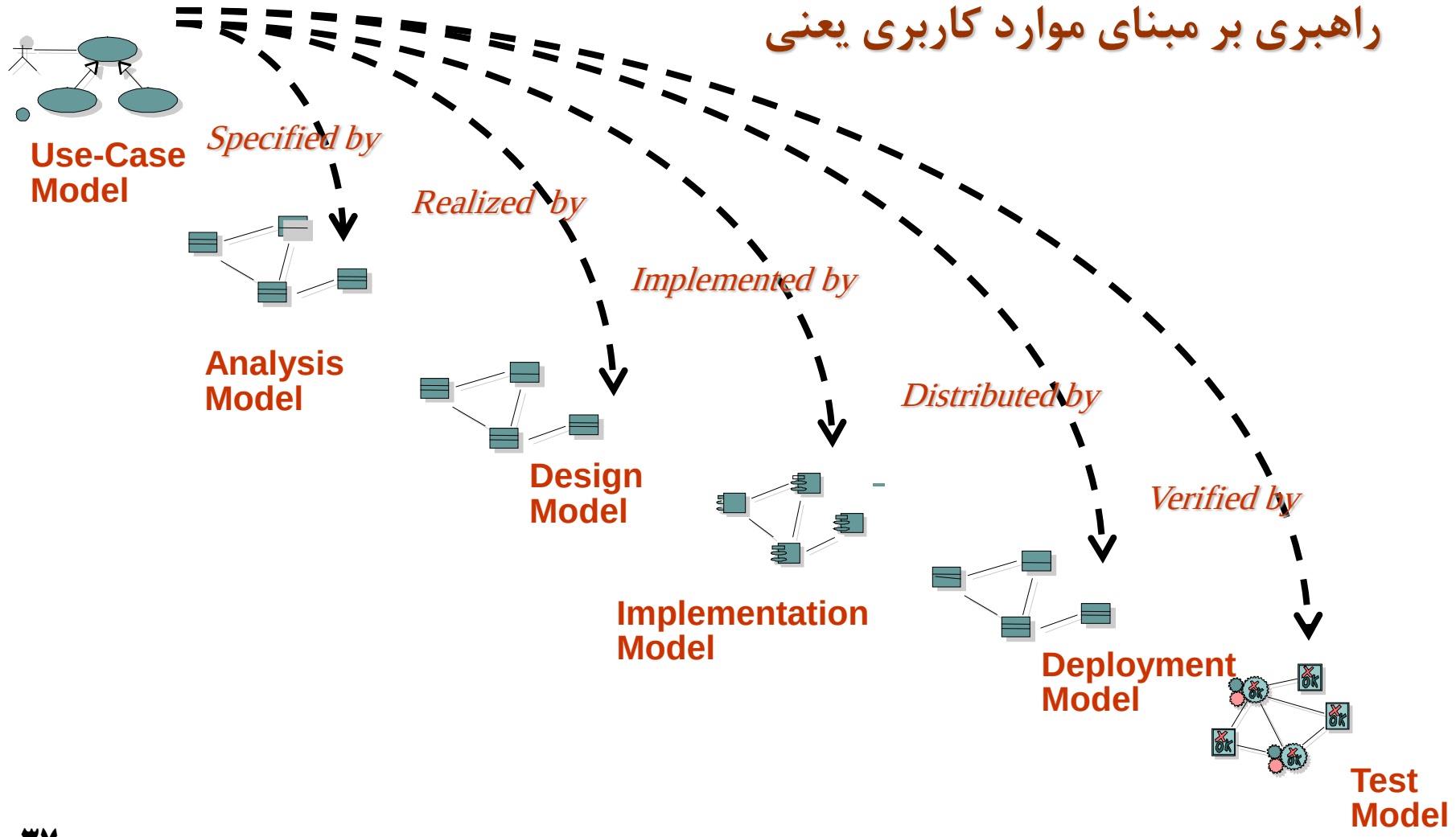
راهبری بر مبنای موارد کاربری

- مورد کاربری عبارت از دنباله‌ای از عملیات است که یک سیستم انجام می‌دهد تا یک نتیجه قابل مشاهده و ارزشمند برای کاربر فراهم نماید

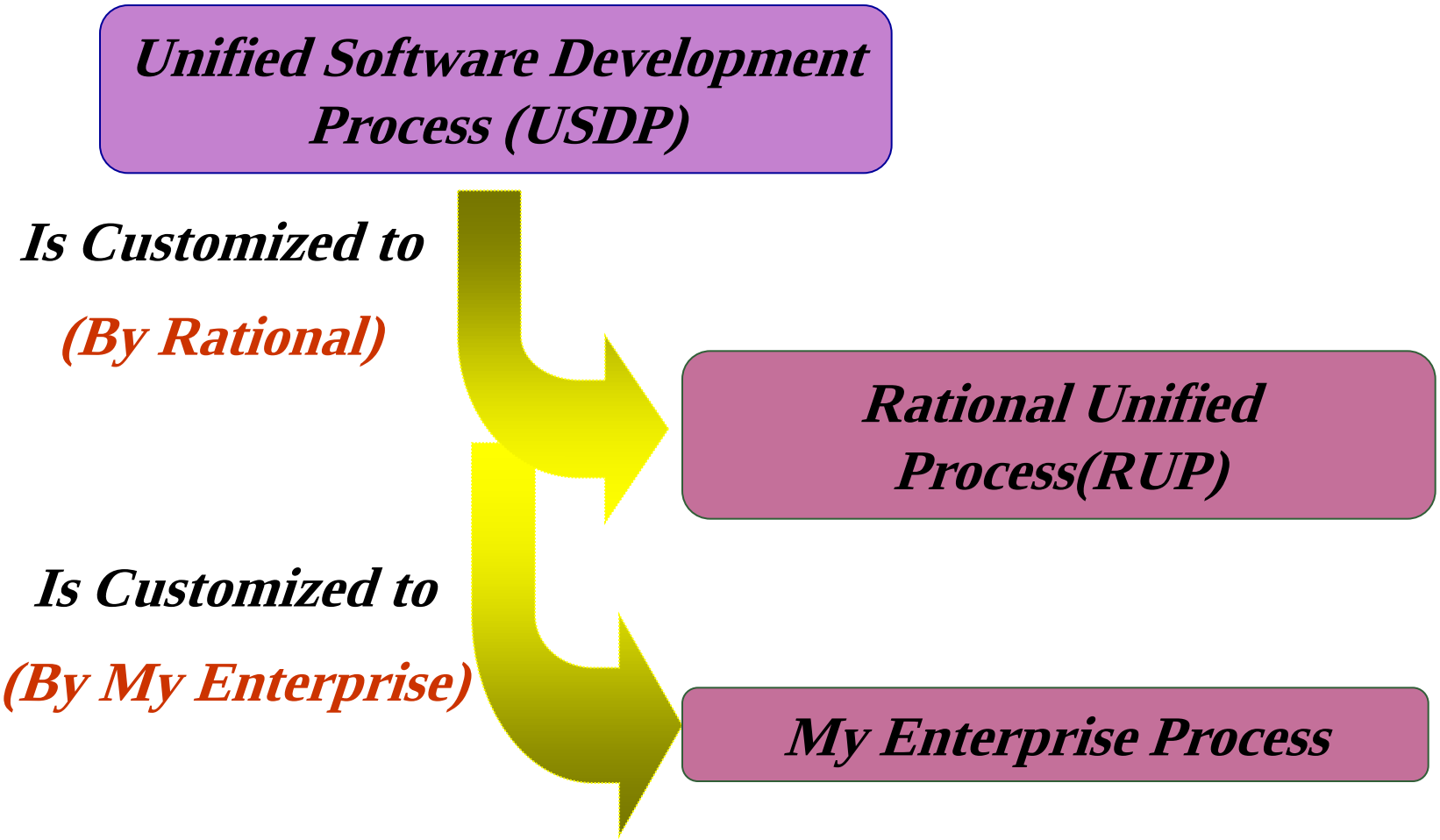
در نگرش سنتی ← سیستم باید چه عملکردهایی از خود را نشان دهد؟

در نگرش مبتنی بر موارد کاربری ← سیستم، به ازای هر کاربر، باید چه عملکردهایی از خود را نشان دهد؟

راهبری بر مبنای موارد کاربری (ادامه)

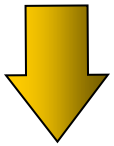


RUP به عنوان یک فرآیند



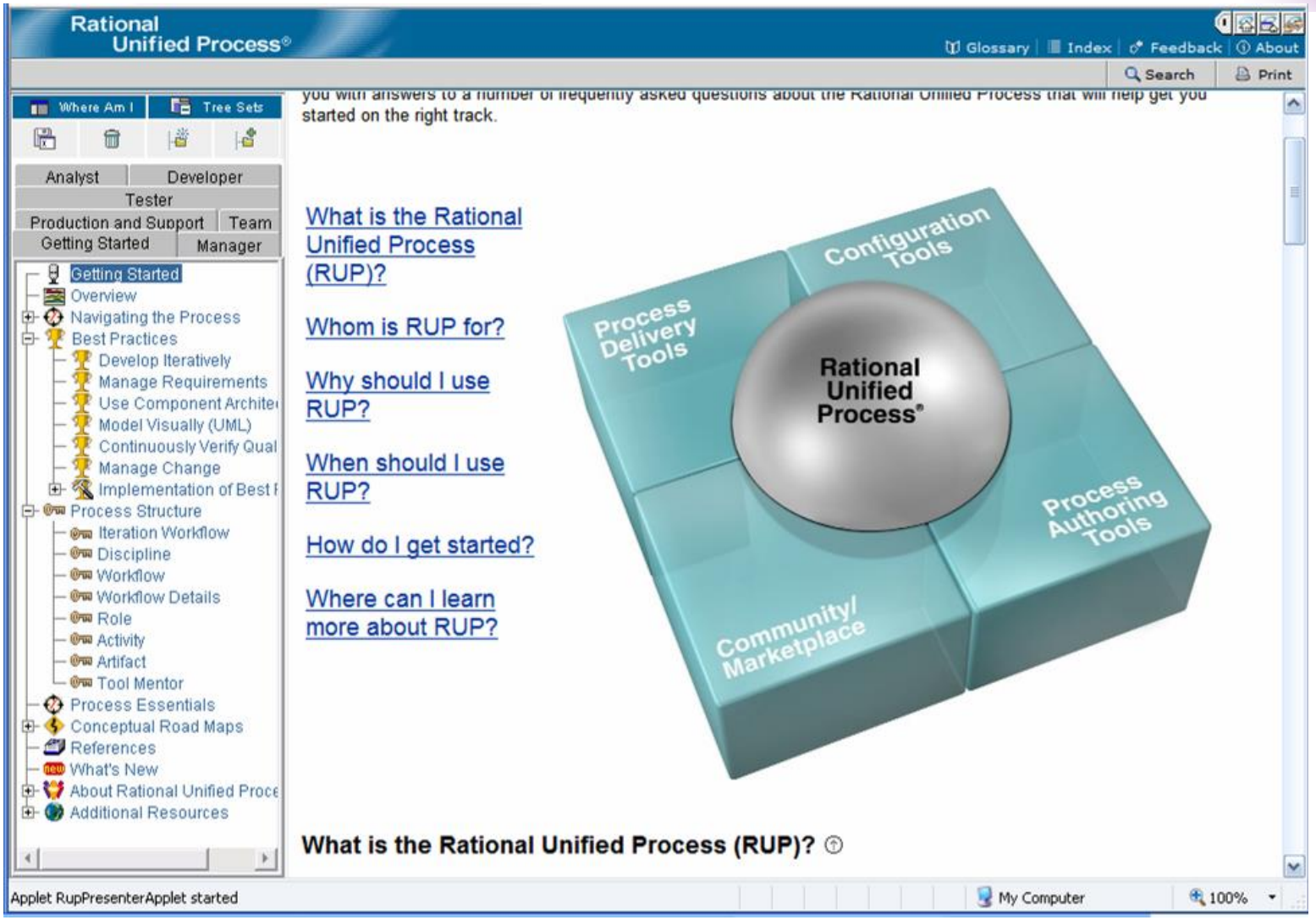
RUP بعنوان یک محصول

- چون مستندسازی RUP بوسیله اسناد کاغذی میسر نیست از اسناد HTML استفاده شده است



- راهنماهای لازم برای بکارگیری RUP به عنوان یک فرآیند تولید در مراحل مختلف تولید نرم‌افزار
- راهنماهای ابزار (Tools Mentors)
- الگوها (Templates)
- یک Development Kit که چگونگی تغییر، گسترش و تنظیم ویژه RUP را نشان می‌دهد

RUP بعنوان یک محصول (ادامه)

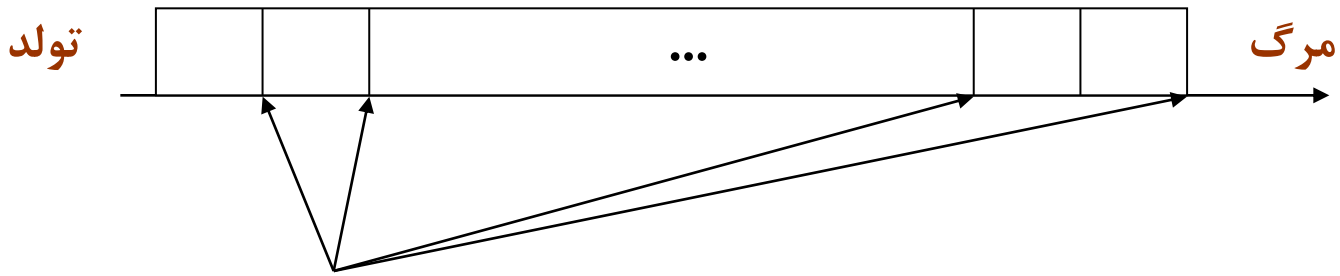


معرفی ابعاد فرآیند RUP

- می‌توان نگاه‌های ذیل را به فرآیند داشت:

۱) جعبه سیاه (*Black Box*)

۲) با توجه به مدت کارکرد مفید سیستم نرم‌افزاری



نسخه‌ها یا نشرها (*Releases*) مختلف

معرفی ابعاد فرآیند RUP (ادامه)

- نرم‌افزار یک محصول فیزیکی که یک بار تولید و مستهلک می‌شود نیست، بلکه مانند یک موجود زنده‌ای که در سازمان تولد و رشد کرده و در دوران حیات خود باید با تغییر نیازهای سازمان و اهداف و ماموریت‌های آن خود را تطبیق دهد

معرفی ابعاد فرآیند RUP (ادامه)

- مثال: زندگی انسان!
  - تولد و مرگ دارد

می‌توان دوران حیات نرم‌افزار را به زندگی انسان تشبیه کرد!!!

  - در میان دوره‌ها نیازهای مشترکی زیادی وجود دارد
  - ولی شیوه برآوردن آنها از یکدیگر متفاوت است
  - همچنین میزان تاکید روی هرکدام متفاوت است

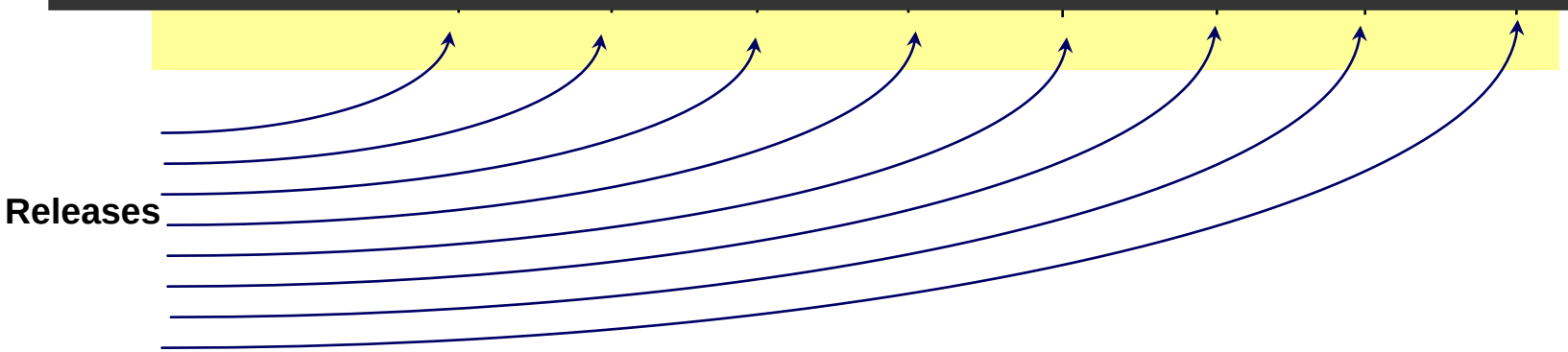
معرفی ابعاد فرآیند RUP (ادامه)

- دوران حیات یک سیستم نرم‌افزاری شامل دوره‌های گوناگونی است
- میان آنها فعالیت‌های مشترکی وجود دارد
- در هر دوره به اندازه معینی روی هر کدام از این فعالیت‌ها تاکید می‌شود
- هرکدام از این دوره‌ها می‌توانند فعالیت‌های ویژه خود را داشته باشند

معرفی ابعاد فرآیند RUP (ادامه)

در RUP دوران حیات یک نرم‌افزار به چهار مرحله آغازین، تشریح، ساخت و انتقال تقسیم می‌شود

سه مرحله اول شامل فعالیت‌های تولید یا توسعه نرم‌افزار بوده و مرحله چهارم دربردارنده انتقال نرم‌افزار به محیط واقعی و نگهداری آن است



معرفی ابعاد فرآیند RUP (ادامه)

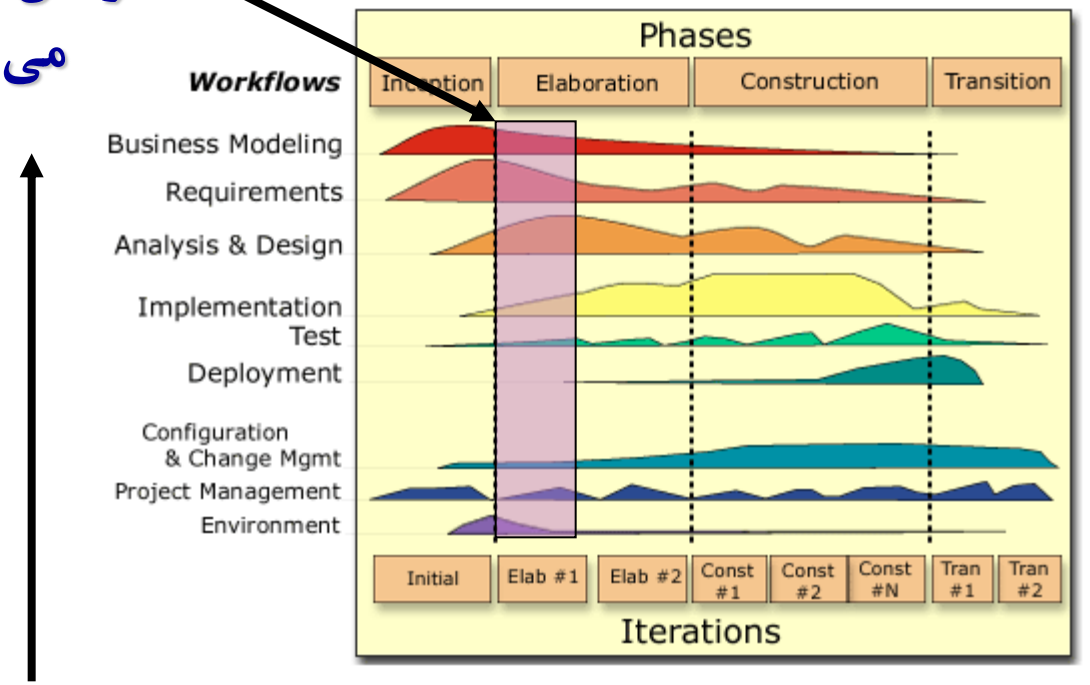
۳) دیدگاه دو بعدی

- RUP یک فرآیند تولید دو بعدی است (برخلاف فرآیندهای تولید سنتی که یک بعدی هستند)
- این ابعاد عبارتند از:
  - بعد (محور) عمودی: گردش کارهای اصلی را نشان می‌دهد
  - بعد (محور) افقی: ساختار چرخه تولید نرم‌افزار در RUP در بستر زمان را نشان می‌دهد

معرفی ابعاد فرآیند RUP (ادامه)

فرآیند دو بعدی

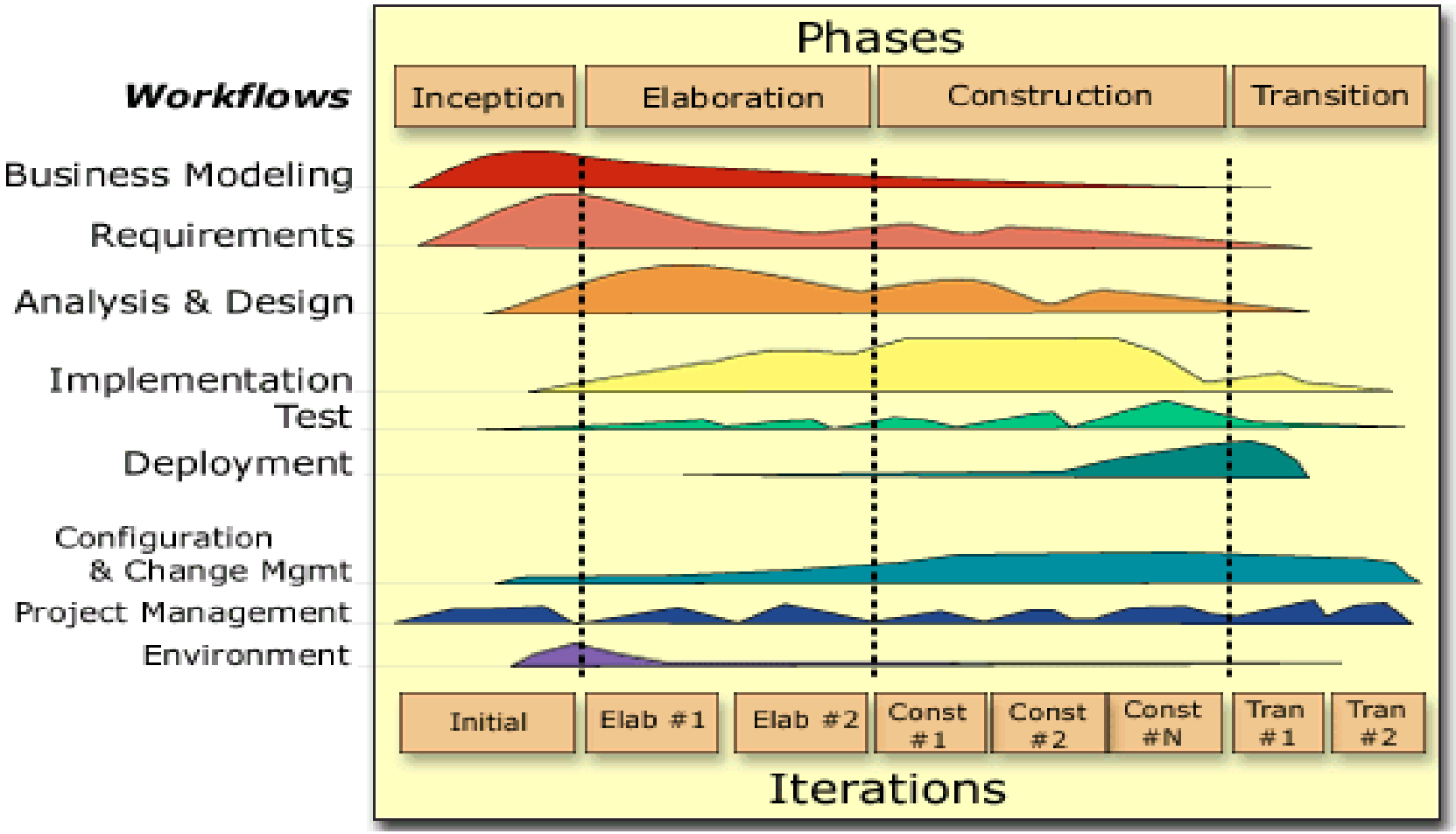
در یک تکرار همه گردش کارها اجرا می شوند



فعالیت‌های مشترک

زمان

معرفی ابعاد فرآیند RUP (ادامه)



پرسش و پاسخ