FDD, LSD

Elizabeth Varghese m

- Feature-Driven Development (FDD)

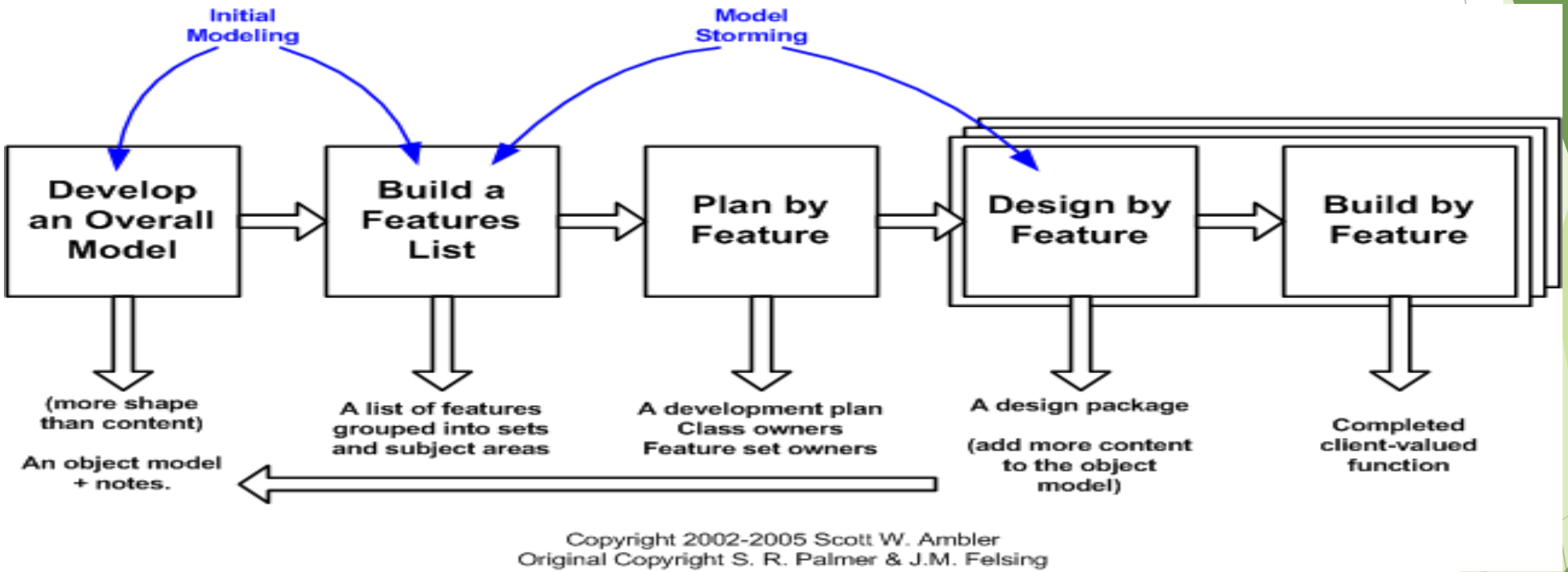
- FDD is a client-centric, architecture-centric, and pragmatic software process
- FDD was first introduced to the world in 1999 via the book Java Modeling In Color with UML

- **How is FDD Different from Scrum?**
- *"During FDD, a feature should be delivered every 2-10 days – which differs from Scrum, in which sprints typically last two, but sometimes four, weeks."*

How Does FDD Work?

- Typically used in large-scale development projects, five basic activities exist during FDD:
- Develop overall model
- Build feature list
- Plan by feature
- Design by feature
- Build by feature

FDD


Initial Modeling
Model Storming
Develop an Overall Model
Build a Features List
Plan by Feature
Design by Feature
Build by Feature
(more shape than content)
An object model + notes.
A list of features grouped into sets and subject areas
A development plan
Class owners
Feature set owners
A design package
(add more content to the object model)
Completed client-valued function
Copyright 2002-2005 Scott W. Ambler
Original Copyright S. R. Palmer & J.M. Felsing

- Develop an overall model:
- Domain and development team members work together under the guiding hand of an experienced Chief Architect.
- Domain Members perform an initial high level walkthrough of the scope of the system and its context.
- Then the domain members perform more detailed walkthroughs of each area of the problem domain.
- After each walkthrough, the domain and development members work in small groups to produce object models for that area of the domain.
- Each small group composes its own model in support of the domain walkthrough and presents its results for peer review and discussion.
- One of the proposed models or a merge of the models is selected by consensus and becomes the model for that domain area.
- The domain area model is merged into the overall model. adjusting the model shape as required.

- Build a features list:
- Based on the partition of the domain by the Domain Experts ,the team breaks the domain into a number of areas (major feature sets).
- Each area is further broken into a number of activities (feature sets).
- Each step within an activity is identified as a feature.
- The result is a hierarchically categorized features list.

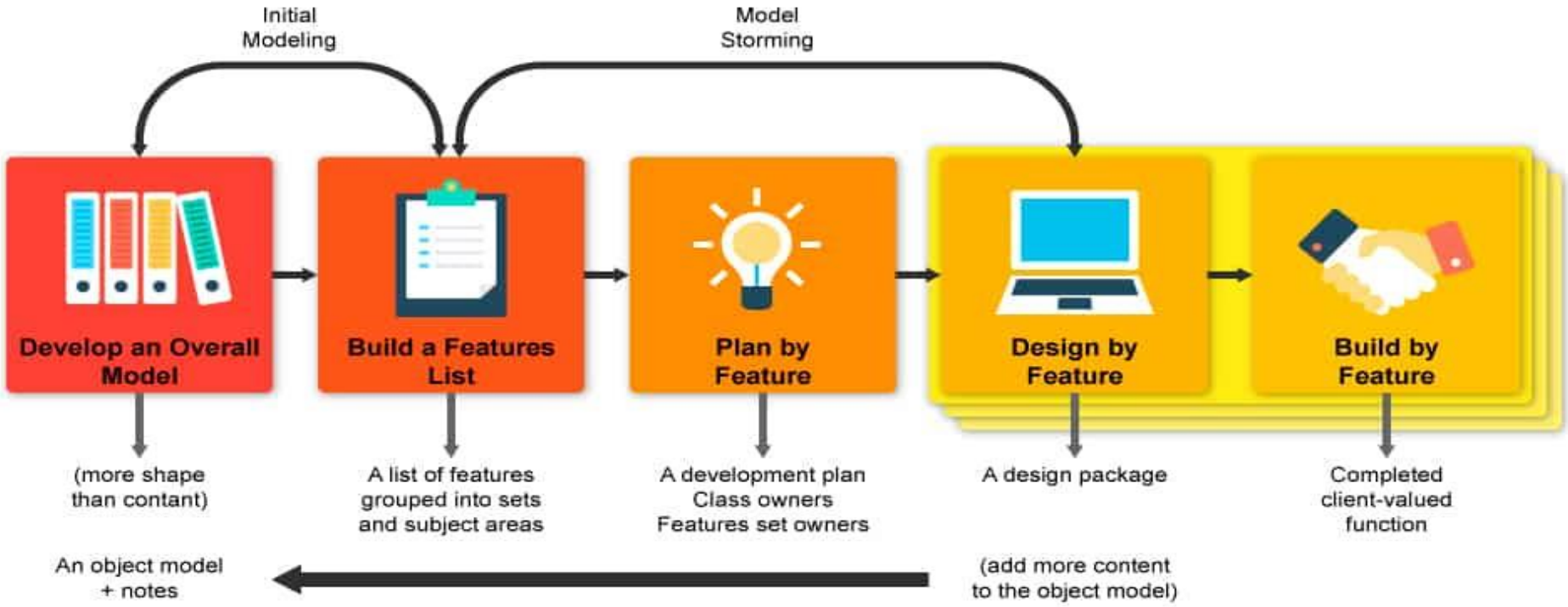
- Plan by feature:
- The project Manager, Development Manager, and Chief Programmers plan the order that the features are to be implemented, based on feature dependencies,load across the development team, and the complexity of the features to be implemented.
- Typical style of refinement is possible here.

Design by feature:

- A number of features are scheduled for development by assigning them to a Chief Programmer.
- The Chief Programmer selects features for development from his or her "inbox" of assigned features.
- Chief Programmer schedules small group of features at a time for development.
- The Chief Programmer then forms a feature team by identifying the owners of the classes, (developers) likely to be involved in the development of the selected features.
- The Chief Programmer then refines the object model based on the content of the sequence diagram

- Build by feature
- Working from the design package produced during the Design by Feature process, the class owners implement the items necessary for their class to support the design for the feature(s) in the work package.
- The code developed is then unit tested and code inspected, the order of which is determined by the Chief Programmer. After a successful code inspection, the code is permitted to build.

FDD


Initial Modeling
Model Storming
Develop an Overall Model
Build a Features List
Plan by Feature
Design by Feature
Build by Feature
(more shape than contant)
A list of features grouped into sets and subject areas
A development plan
Class owners
Features set owners
A design package
Completed client-valued function
An object model + notes
(add more content to the object model)

- Lean Software Development (LSD)

- Lean Software Development (LSD) is an agile framework based on optimizing development time and resources, eliminating waste, and ultimately delivering only what the product needs.
- The Lean approach is also often referred to as the Minimum Viable Product (MVP)strategy, in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don't like and want to be added, and then iterates based on this feedback.
- Adapted from the Toyota Production System

lean manufacturing principles

- Lean development can be summarized by seven principles, very close in concept to :
- Eliminate waste
- Amplify learning
- Decide as late as possible
- Deliver as fast as possible
- Empower the team
- Build integrity in
- Optimize the whole

Agile Lean Software Development

Agile modeling (AM)

- It is a methodology for modeling and documenting software systems.
- It is a collection of values and principles, that can be applied on an (agile) software development project.
- This methodology is more flexible than traditional modeling methods, making it a better fit in a fast changing environment. It is part of the agile software development tool kit.
- Agile modeling is a supplement to other agile development methodologies such as Scrum, extreme programming (XP), and Rational Unified Process (RUP). It is explicitly included as part of the disciplined agile delivery(DAD) framework.

core practices:

- There are two core practices:
- **Documentation**
- Document continuously. Documentation is made throughout the life-cycle, in parallel to the creation of the rest of the solution.
- Document late. Documentation is made as late as possible, avoiding speculative ideas that are likely to change in favor of stable information.
- Executable specifications. Requirements are specified in the form of executable "customer tests", instead of non-executable "static" documentation.
- Single-source information. Information (models, documentation, software), is stored in one place and one place only, to prevent questions about what the "correct" version / information is.

- **Modeling**
- **Active stakeholder participation**. Stakeholders of the solution/software being modeled should be actively involved with doing so. This is an extension of the on-site customer practice from Extreme Programming.
- **Architecture envisioning.** The team performs light-weight, high-level modeling that is just barely good enough (JBGE) at the beginning of a software project so as to explore the architecture strategy that the team believes will work.
- Inclusive tools. Prefer modelling tools, such as whiteboards and paper, that are easy to work with (they're inclusive).
- **Iteration modeling**. When a requirement/work item has not been sufficiently explored in detail via look-ahead modeling they team may choose to do that exploration during their iteration/sprint planning session. The need to do this is generally seen as a symptom that the team is not doing sufficient look-ahead modeling.

Bibliography

- ASD: https://www.tutorialspoint.com/adaptive_software_development/adaptive_software_development_lifecycle.htm
- DSDM

https://www.solutionsiq.com/agile-glossary/dynamic-systems-development-method-dsdm/

https://en.wikipedia.org/wiki/Dynamic_systems_development_method

- FDD

https://www.productplan.com/glossary/feature-driven-development/

http://agilemodeling.com/essays/fdd.htm#:~:text=Feature%20Driven%20Development%20(FDD)%20and%20Agile%20Modeling&text=Feature%2DDriven%20Development%20(FDD),Programming%20(XP)%20calls%20customers.