Merging Systems Engineering Methodologies with the Agile Scrum Framework

NUWC DIVISION, KEYPORT

C/40

Fleet Readiness Department

Dr. Dallas Rosson

Principal Engineer
Digital Transformation Division
Naval Undersea Warfare Center, Detachment Keyport

(w) (360) 315-0702
dallas.j.rosson.civ@us.navy.mil

Biography

- **Current duties:**
  - Naval Undersea Warfare Center Division Keyport,
    C/44 Digital Transformation Division, Principal Engineer
  - Office of the Undersecretary of Defense,
    Chief Architect, Enterprise Part Management System
  - University of Washington, Professor of Practice, Systems Engineering

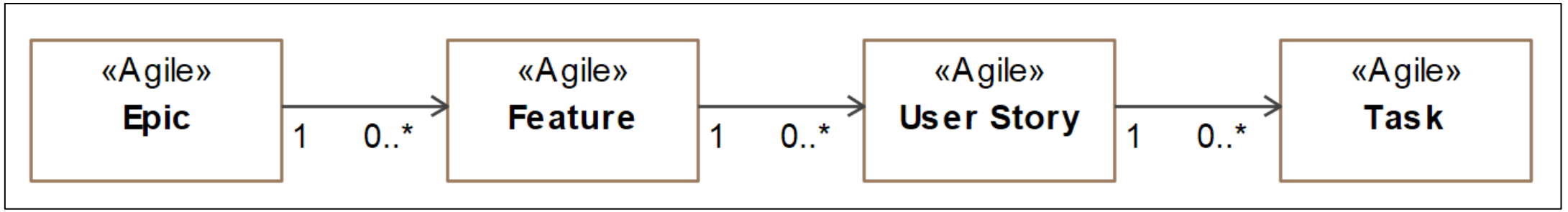
- **Education:**
  - Doctor of Engineering, Systems Engineering, Colorado State University
  - Master of Science, Computer Science, University of Washington
  - Master of Science, Systems Engineering, George Washington University
  - Certifications: PMP, Advanced Certified Scrum Master,
    Advanced Certified Scrum Product Owner

Background

Why do we need this?

- **Agile projects are often unsuccessful and/or failures.**
  - 46% of customers report projects developed under an Agile umbrella as “unsuccessful” within the boundaries of client benefits, cost control, and time control [1].
- **Agile projects often utilize Connextra formatted User Stories, which are not requirements, but rather goals.**
  - User Stories are too high level to detail customer requirements.
  - Difficult for V&V activities to fully certify that all requirements are being met.
  - Rely on a lot of guesswork from developers rather than detailing customer understanding.
- **Modeling strategies and techniques increase system domain knowledge and understanding, lowering risks of mis-development which can have an adverse affect on cost, schedule, and performance.**
  - Developers surveyed report that modeling positively affected project development [2].
  - Increases understanding of requirements traceability throughout the system.
  - Increases ease of communication of system capabilities with stakeholders.
  - Meets requirements for DoDAF and AAF models usually without having to generate additional content.

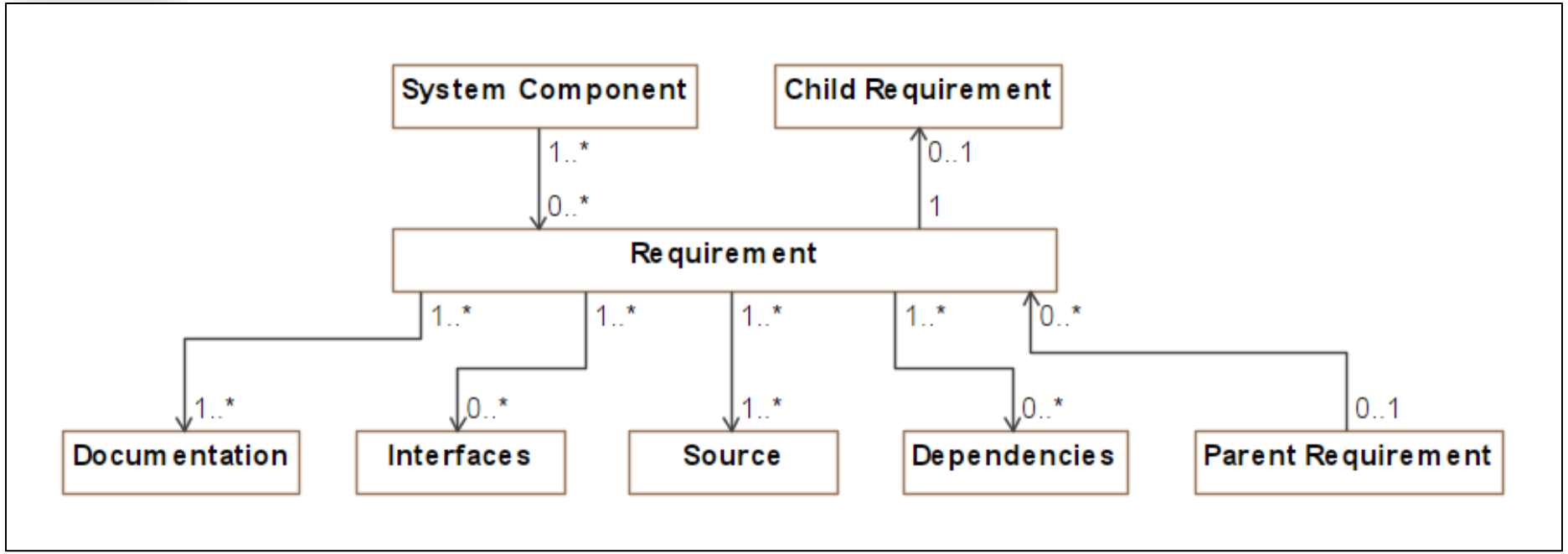
User Stories versus INCOSE Requirements



User Story Traceability:

- Hierarchical “one-to-many”.
- Not mentioned in the Scrum Guide, Agile Software Development, or Agile Practice Guide [3][4][5].
- Utilization outside of hierarchical definition is not standardized in the software development industry.
- The simple structure prevents a full understanding of the domain and system functions and relations [6].
- Not detailed enough to meet INCOSE requirements standards.

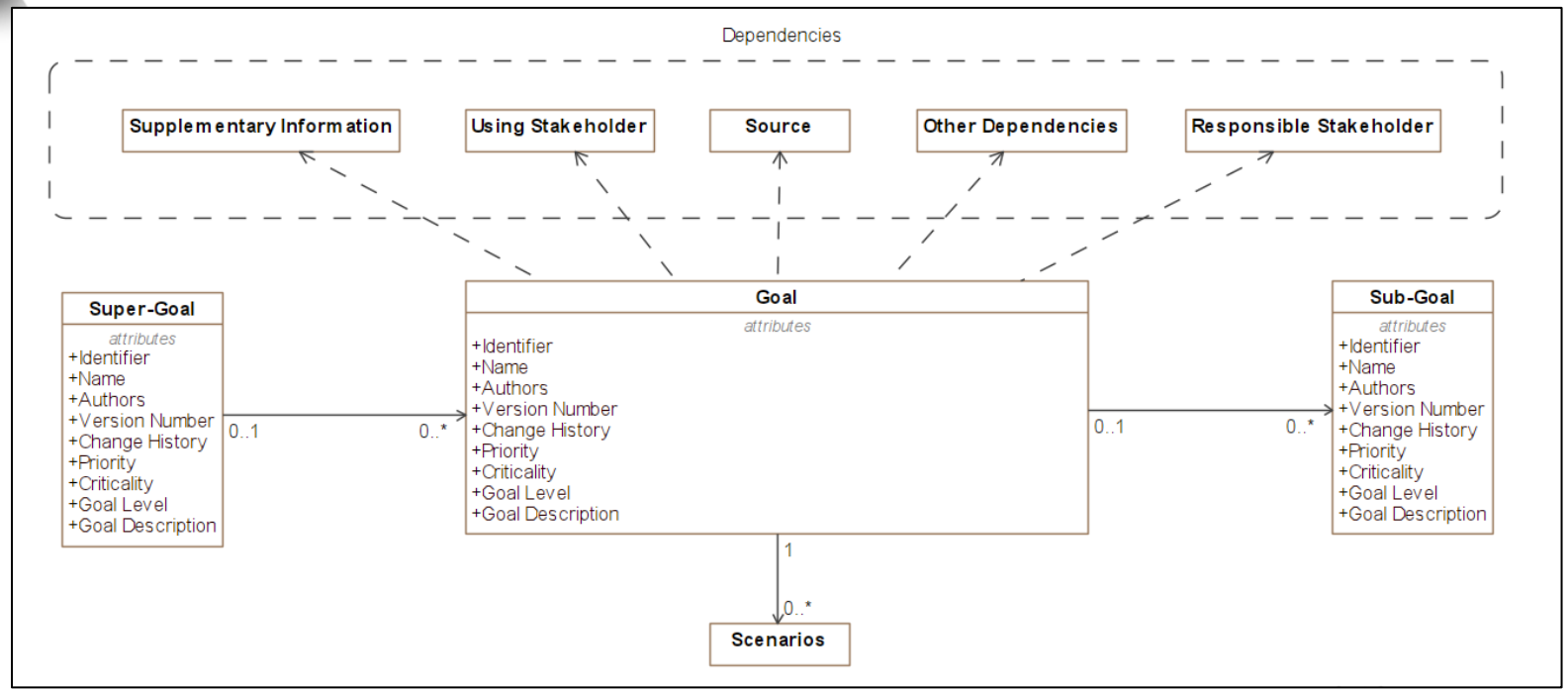
User Stories versus INCOSE Requirements



INCOSE Requirement Traceability:

- Full system traceability allowing system developers to know the identity, location, relationships, pedigree, origin of data, materials, and parts of all system elements [7].
- Traced both vertically, or hierarchically, and horizontally, or at system and life cycle levels [8].
- Tracing requirements back through derivations, sources, interfaces, documentation, and many other inputs allows systems engineers to fully conceptualize the problem space, building a mental and real model of not only what needs to be built, but also the how and why each requirement is necessary.

User Stories versus INCOSE Requirements



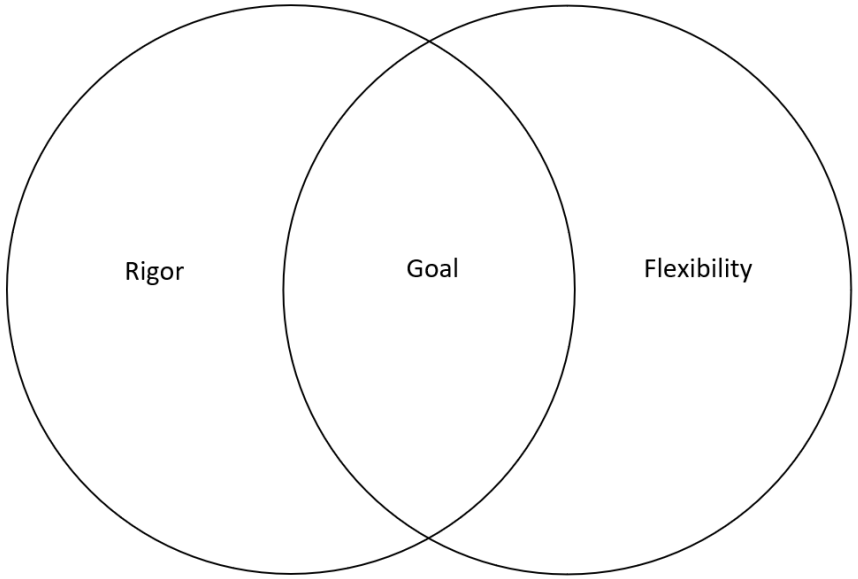
User Stories are Goals not Requirements:

- ISO/IEC/IEEE 29148-2018 states the following about goals, “The term ‘Goal’ (sometimes called ‘business concern’ or ‘critical success factor’) refers to the overall, high-level objectives of the system.
- User Stories are not detailed enough to be considered true requirements, as they do not have all the metadata associated with an INCOSE requirements management style requirement.
- The simple structure of the Connextra User Story format prevents a full understanding of the domain and system function and relations to be drawn from a User Story.

Is there a solution?

Create a methodology that incorporates systems engineering methods while embracing the flexibility of Agile Scrum:

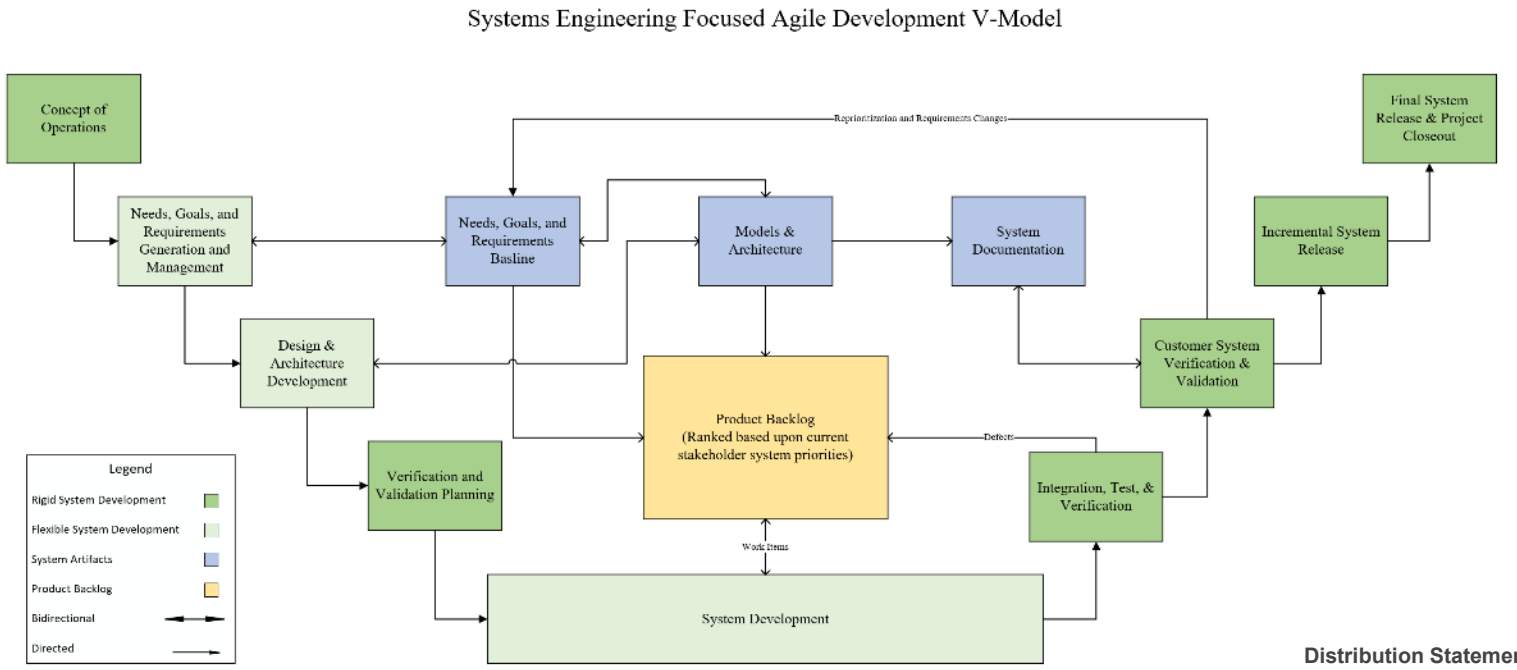
- Must focus on direct customer feedback.
- Baselines, but updates.
- Model Based Systems Engineering integration.
- Must meet Adaptive Acquisition Framework requirements for all pathways.
- Focus on building the right thing, the right way, while minimizing risk to the stakeholder and the developer equally.
- Documentation and technical rigor cannot be ignored!



Systems Engineering Focused Agile Development (SEFAD)

Developed to apply systems engineering and technical rigor to Agile Scrum execution in a thoughtful way.

- Implement rigor where it brings value.
- Does not require rigor for rigors sake.
- Takes advantage of the flexibility of the Agile Scrum Framework.
- Focuses on customer interaction and documentation of needs and requirements.



Systems Engineering Focused Agile Development

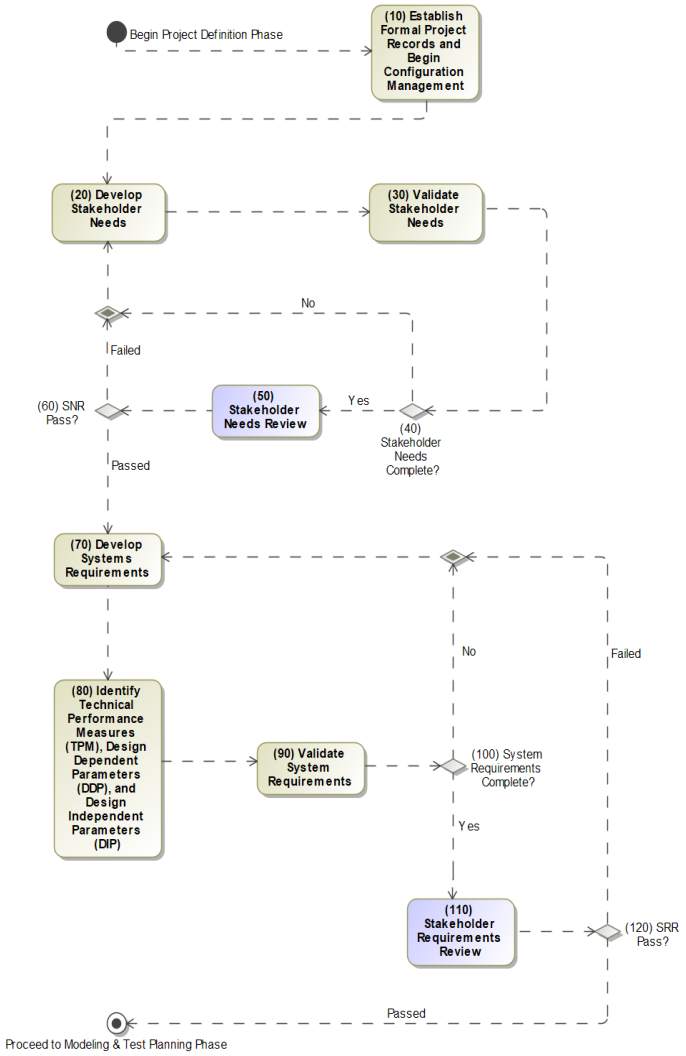
Four Phases

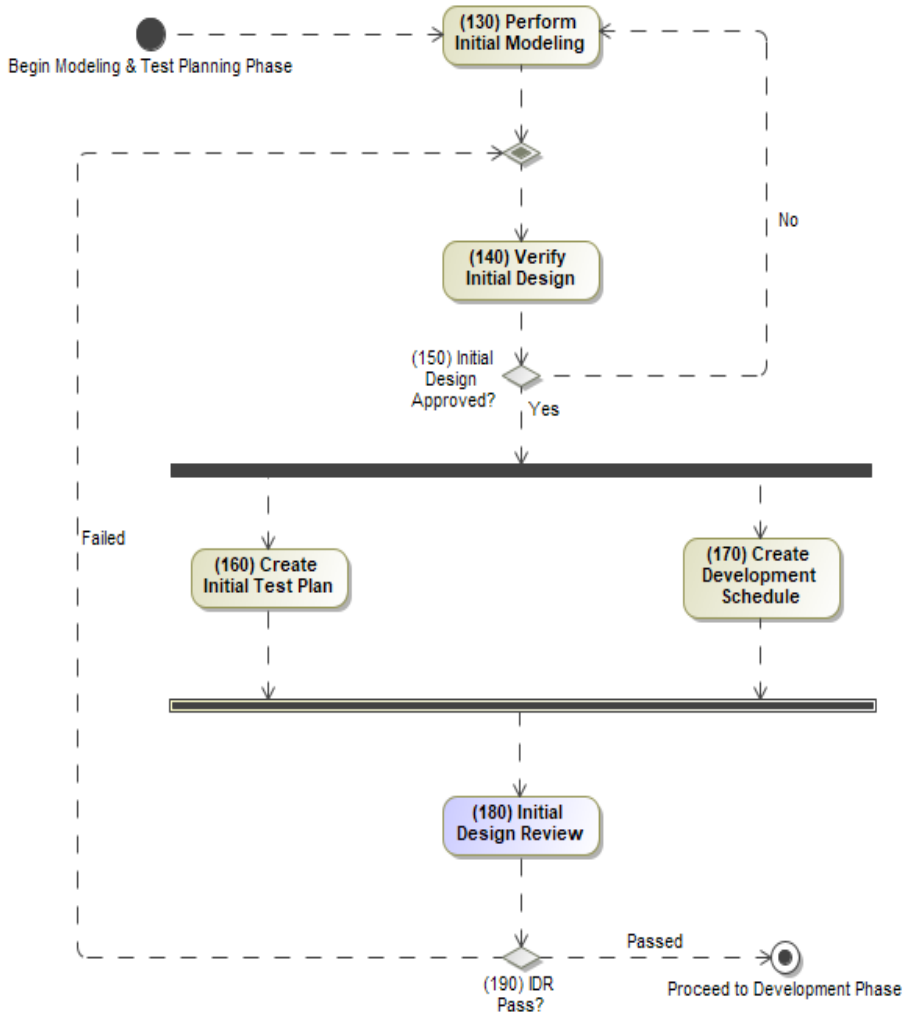
- Project Definition Phase
- Modeling & Test Planning Phase
- Development Phase
- Project Finalization Phase

Each phase is focused on delivering quality products to the customer, whether this is a requirements document or a finished software product.

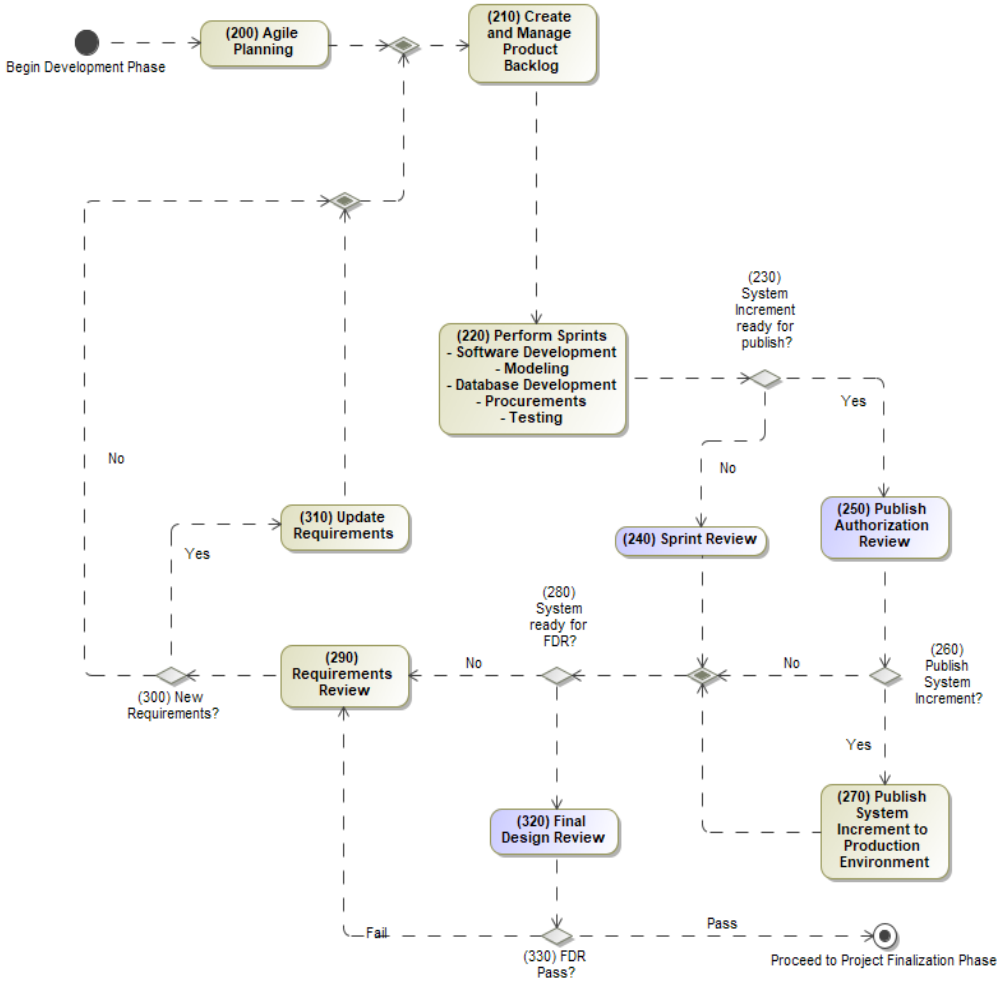
- All phases utilize iterative loops and frequent customer interaction for increased feedback.
- Encourages early looks at deliverables to prevent schedule slippage through pass/fail gates.
- Nothing is fully baselined and “locked in” until the Project Finalization Phase to ensure flexibility and capability of pivoting to changing customer needs.

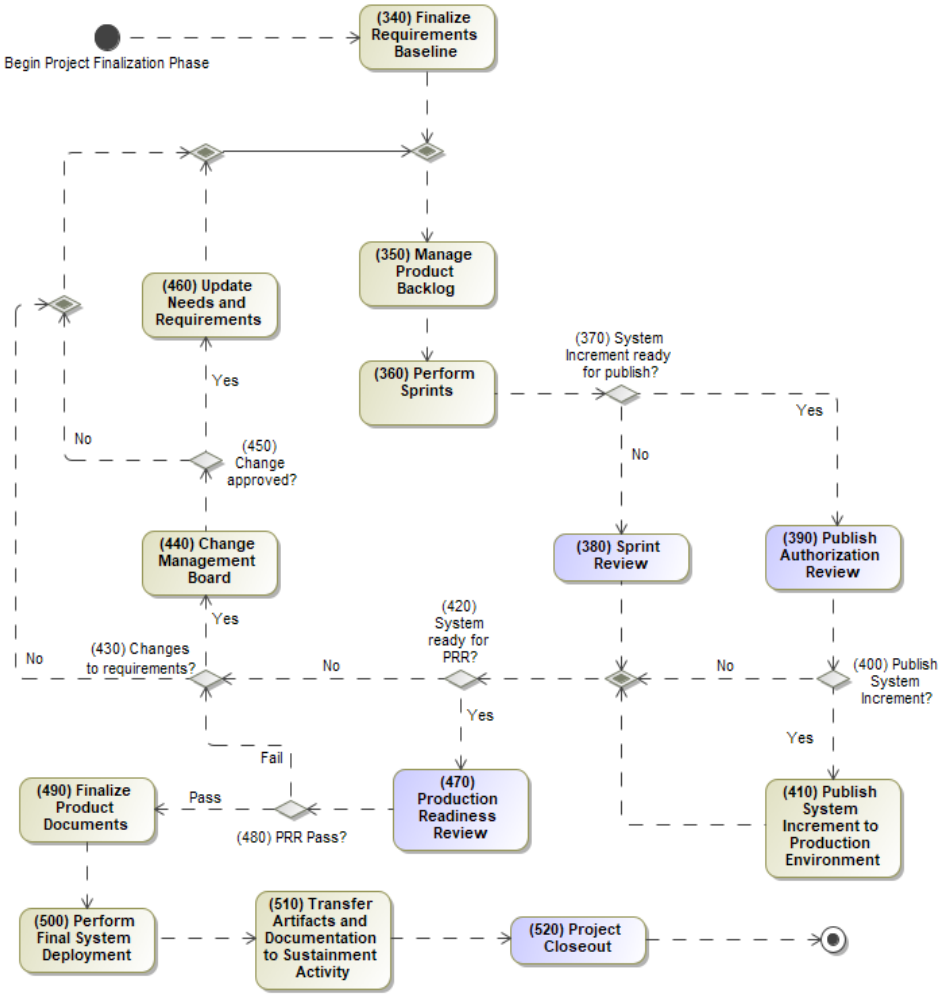
Project Definition Phase and Modeling & Testing Phase





Development Phase and Product Finalization Phase





Results of Implementing this Method - OMIS

Conducted a survey of stakeholders of the Obsolescence Management Information System after implementing systems engineering methods in the development process.

- Implemented Requirements Management and MBSE.
- Results were overwhelmingly positive versus previous years.
- Noted a 300% reduction in defects generated.

Question	Average Result	Notes
Prefer user stories or shall statements?	Shall Statements	83% Prefer Shall Statements, 17% Prefer User Stories
Are you more or less confident in the quality of the OMIS System?	Equal/More	50% As Confident, 50% More Confident
Do you feel prototypes or wireframes have a positive impact in understanding and communicating user needs?	Yes	100% Positive Impact
In the last two years, do you feel you are finding more or less bugs/defects in the system than in years prior?	Less	100% Feel Less Bugs/Defects
Overall, how satisfied are you with the OMIS development process as opposed to years prior? (On a scale of 1-10, 1 being unsatisfied and 10 being very satisfied)	8	Average calculated based upon all respondents
Do you feel communication and development status has become more transparent or less in the last two years?	More Transparent	67% Improved Communication, 33% Equal to Previous Years

Results of Implementing this Method – S2DE

Background:

- S2DE had not successfully passed a Test Readiness Review (TRR) to make a major release in over 2 years.
- The system requires testing in representative environments prior to release.
- Few requirements had been documented and were at a very abstract level.
- Utilized Agile Scrum for development.

Action:

- Implemented Requirements Management.
- Implemented rigorous Test Management with all test cases traced directly to requirements.
    - Modeled test case traceability for better understanding and visualization.
- Increased customer collaboration.

Results:

- Successfully held and passed a TRR.
    - Was able to report full system test coverage through formalized test management, planning, and visual traceability.
- Successfully held a Navy wide test event, resulting in passing a Production Readiness Review (PRR) for a major version.
- Of special note, the software development team reported higher job satisfaction, less stress, and an increased understanding of the system as a whole.

Results of Implementing this Method – EPMS

Background:

- New Adaptive Acquisition Framework (AAF) cross-DoD logistics system.
- Originally contracted as a Major Acquisition Pathway.
- Contractors given a few pages of User Stories as requirements.

Action:

- Worked closely with contractors to convert User Stories to fully traced requirements.
  - Kept User Stories for software developers to work with but included traceability to requirements.
- Utilized SysML to model the domain, context, requirements, actions, and sequences.
- Updated contracting language to focus on iterative deliverables vice milestone delivery.

Results:

- Converting the User Stories to INCOSE style requirements allowed for a better understanding of the functional needs of the system.
- Tracing requirements to User Stories allowed developers to group requirements together for execution, but also made clear testing requirements and specified customer needs.
- Modeling fueled discussions and clarified systems functionality prior to software development efforts.
  - Models were created to meet DoDAF requirements along with industry standard SysML diagrams.

Current Adopters

Current adopters of this methodology:

- NUWC Division Keyport, Digital Transformation Division
- Puget Sound Naval Shipyard (PSNS) Intermediate Maintenance Facility (IMF), Code 300.1
- NUWC Division Keyport, Undersea Weapons Department, Engineering & Production Enablement
- NUWC Division Keyport, Fleet Readiness Department, Electrical Engineering Applied Technology Branch
- NUWC Division Newport, In Service Engineering Activity 1533
- Naval Surface Warfare Center (NSWC) Crane Division, Microelectronics Assurance Branch (GXVR)
- Bureau of Labor Statistics, Office of Technology and Survey Processing
- Defense Human Resources Activity, Defense Manpower Data Center, Technical Services Division

Feedback and Participation

Ongoing Research:

- Applying SEFAD to projects is an ongoing research project.
- Always looking for volunteers to apply the methodology.

Eliciting Feedback on Successes and Failures in Software Development Projects:

- Gathering data for what works and what doesn't work in DoD software development.
  - Problems with too much rigidity or difficulties due to project structural constraints on execution.
  - Problems with too little structure or being "too Agile".
  - Successes in applying SE methods to software development.
  - Successes in applying Agile methods to software development that was previously too structured or predictive type execution.
- Can be any methodology. This is for learning and documentation.

References

[1]: Jorgensen, Magne, “Do Agile Methods Work for Large Software Projects?”, in Proceedings of the 19th International Conference, XP 2018, “Agile Processes in Software Engineering and Extreme Programming”, May 21-25, 2018, Porto, Portugal

[2]: J. Hutchinson, J. Whittle and M. Rouncefield, "Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure," Science of Computer Programming, vol. 89, pp. 144-161, 2013.

[3]: Project Management Institute, Agile Practice Guide, Project Management Institute, 2017.

[4]: K. Schwaber and J. Sutherland, The Scrum Guide, Ken Schwaber and Jeff Sutherland, 2020.

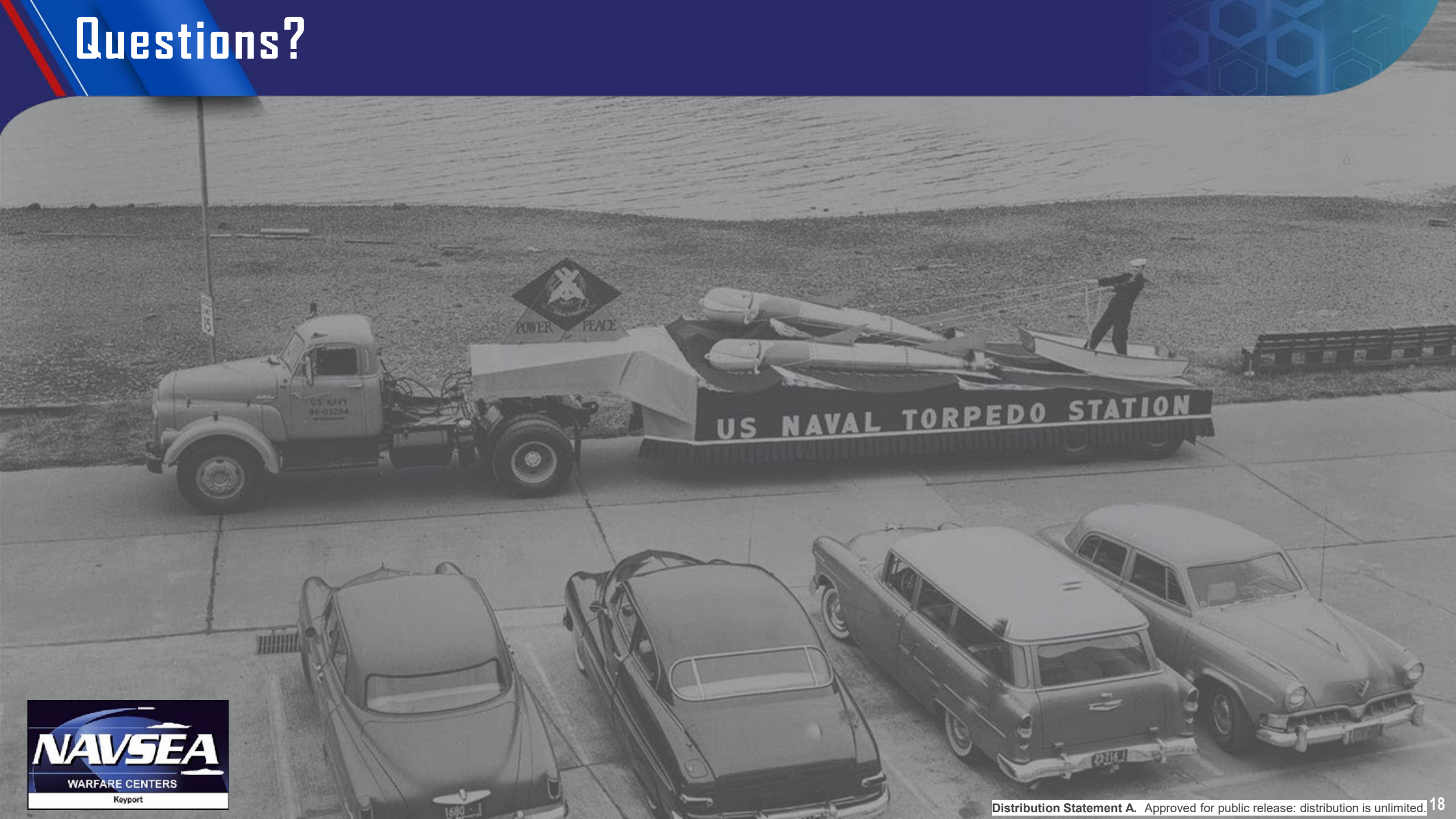
[5]: T. Dingsoyr, T. Dyba and N. B. Moe, Agile Software Development: Current Research and Future Directions, Berlin, Germany: Springer Berlin, Heidelberg, 2010.

[6]: T. Günes and F. B. Aydemir, "Automated Goal Model Extraction from User Stories Using NLP," in 2020 IEEE 28th International Requirements Engineering Conference (RE), Zurich, Switzerland, 2020.

[7]: International Council on Systems Engineering, Systems Engineering Handbook, 5th Edition, San Diego, CA: International Council on Systems Engineering (INCOSE), 2023.

[8]: INCOSE Requirements Working Group, Guide to Writing Requirements, San Diego, CA: International Council on Systems Engineering (INCOSE), 2023.

Questions?



NAVSEA
WARFARE CENTERS
Keyport