

Developing a Product Line Approach for Flight Software

Mike Stark, Dave McComas

NASA/Goddard Space Flight Center

Guilherme H. Travassos

COPPE/Federal University of Rio de Janeiro

Maurizio Morisio

Politecnico di Torino

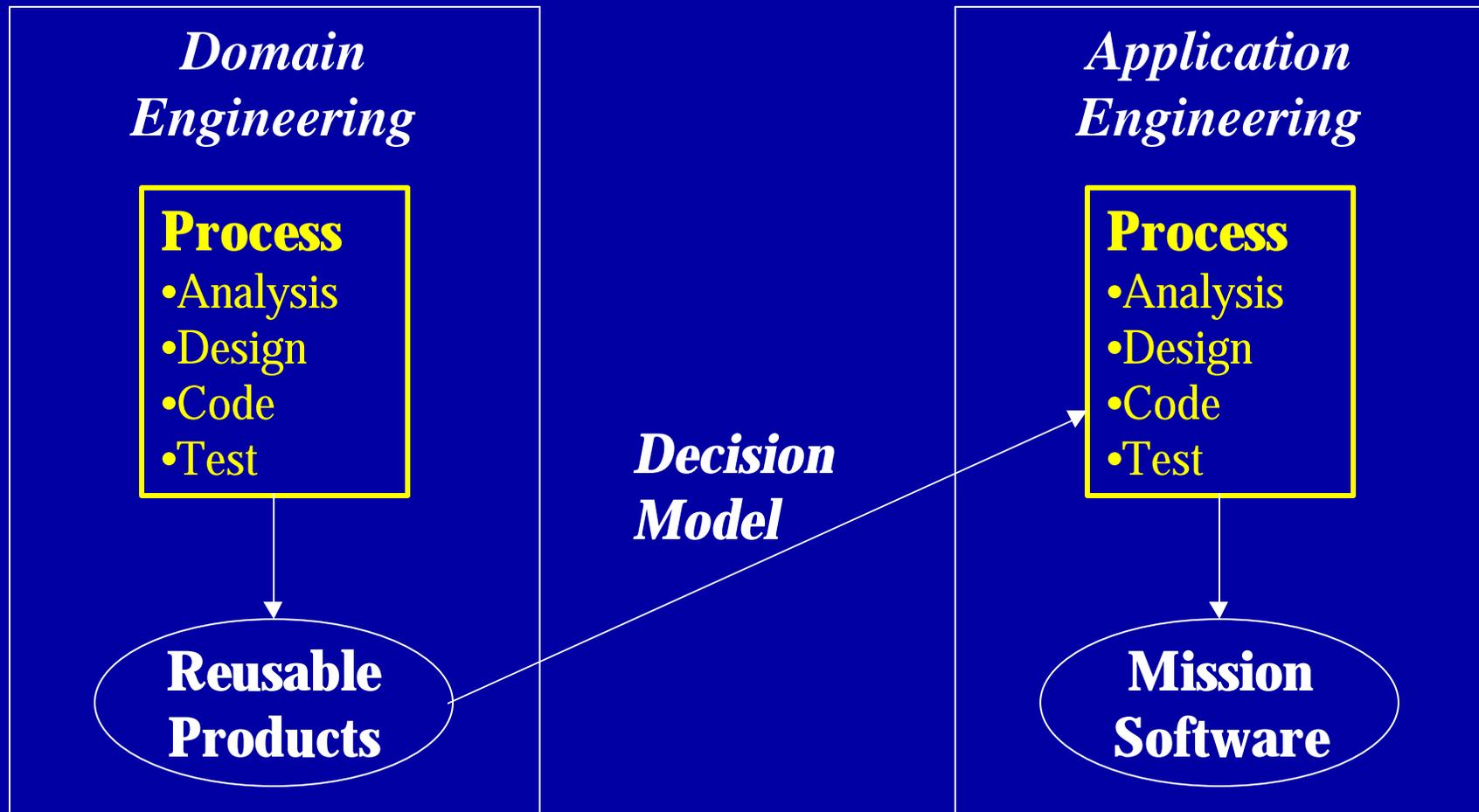
Presentation Outline

- Product line development
- Project background
- Product line development approach
- Observations
- Future work

What's a Product Line?

- “A family of products designed to take advantage of the common aspects and predicted variabilities”
David Weiss, Software Product Line Engineering
- Product Line Methodologies
 - Synthesis (Software Productivity Consortium)
 - Family-Oriented Abstraction, Specification and Translation (FAST) (Weiss and Lai)
 - PuLSE (Fraunhofer Center)
 - FODA (Software Engineering Institute)

Product Line Development



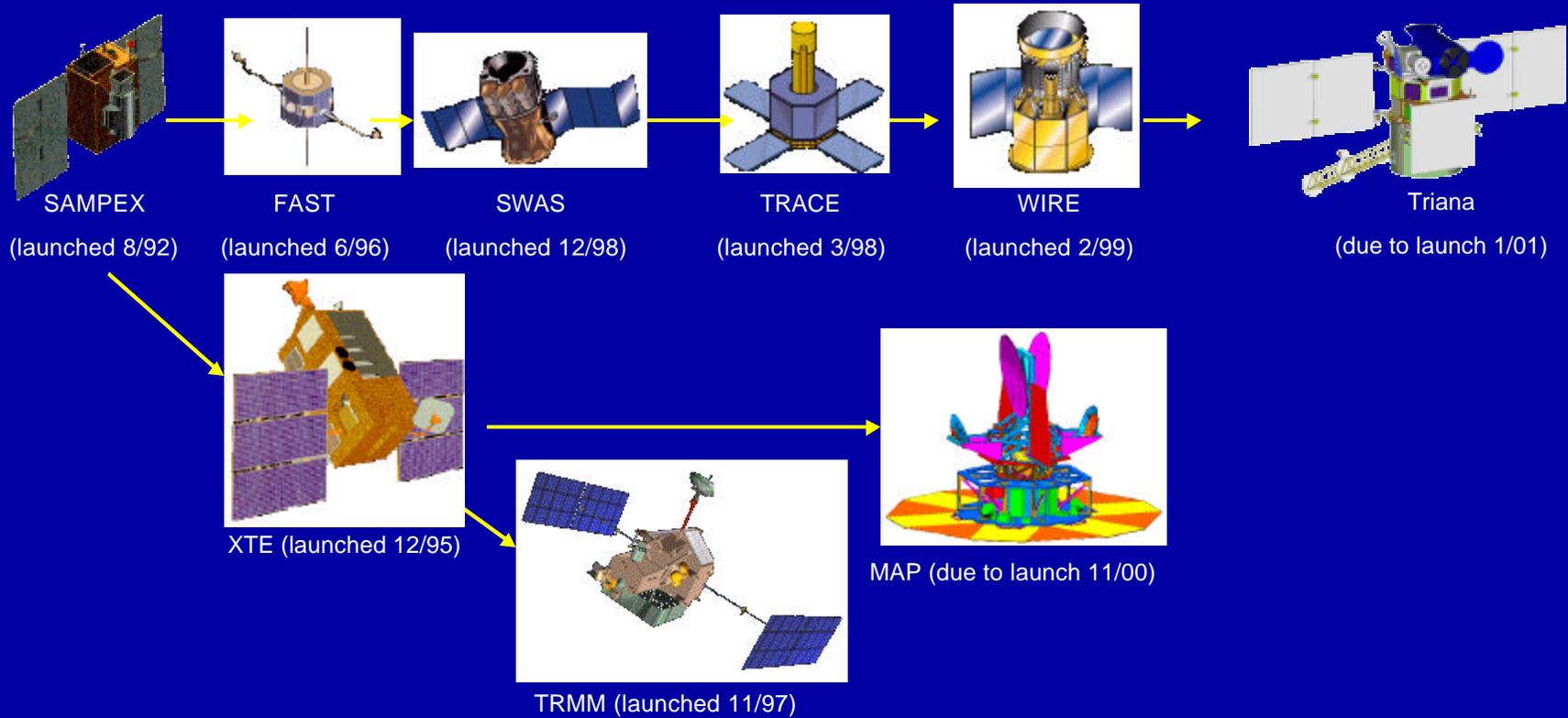
Why a Flight Software Product Line?

- The Flight Software Branch (FSB) was formed as part of a major reorganization at GSFC
 - Personnel come from four predecessor organizations
- Current FSB need:
 - Institutionalize knowledge
 - Reduce development time without increasing risks
 - Control product growth
 - Apply technology consistently
- Our quality comes at a high dollar and people price (informal observation)

Why a Guidance, Navigation and Control (GNC) Product Line?

- Guidance, Navigation and Control (GNC) is
 - Needed for each mission
 - A mature problem domain
- There is a family of similar missions to draw from
- There is experience to build on
 - FSB knowledge of problem and flight architecture
 - SEL experience with process analysis
 - GSS experience developing reusable ground software for GNC problem

Family of supported missions



Product Line Project Goals

- Provide the framework for continual improvement
 - with minimal impact to current development efforts
- Reduce development time without sacrificing quality
- Increase productivity
 - Smaller development teams, or more complex software
- Provide rapid prototyping capability that includes
 - High fidelity models
 - Ability to export code to and import code from analysis tools (e.g. MATLAB)
- ***Question: Can we do it?***

Product Line Project Approach

- Interpreting “Can we do it?”
 - Is approach technically feasible?
 - Is it cost effective?
 - Does it serve the needs of all the stakeholders?
- Development Approach
 - Evaluate existing product line methods
 - Perform high-level domain assessment
 - Implement successively more complex prototypes
 - Produce new domain and application products
 - Define, evaluate, and update process definitions

Evaluate Product Line Methods

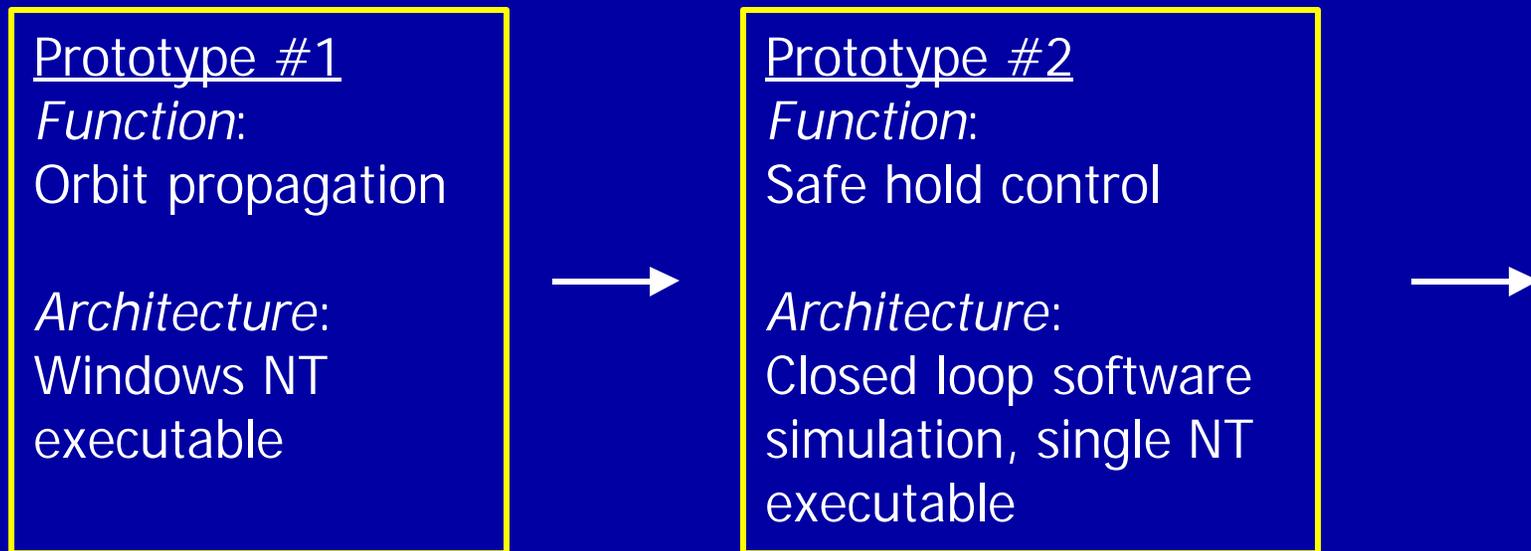
- Compare and contrast existing product line approaches
 - Primarily FAST and Synthesis, due to available training and consulting
 - Now looking at PuLSE approach to domain assessment
- Apply methods during prototyping
- Use GSS experience as reality check

Perform High-level Domain Assessment

- Approach
 - Select a set of reference missions
 - Bound the domain
 - Identify essential subdomains
 - Establish priorities for prototyping
- Notation
 - GNC domain context diagram
 - Subdomain dependency diagram
 - Commonality /variability assumptions

Implement Prototype

- Series of small applications, each of which addresses
 - GNC models (domain functionality)
 - Flight software architecture
 - Product line processes



Process & Product Prototyping

- Synthesis was our starting point
 - Outlined full product line development process
 - Domain analysis is currently most mature part
 - Commonality/variability analysis was most important contribution from Synthesis and FAST approaches
- Additional notation
 - UML extensions representing variabilities
 - Mission/capability matrices
 - Mission/client matrices

Coarse Sun Sensor (CSS) Clients

Capability using CSS model

		Estimation	Control	Mode/Constraint Management, Telemetry
Mission	Wire	<i>Ac_USun_BF</i> • TRIAD • Kalman Filter Residuals	<i>Ac_USun_BF</i> • Digital Sun Control (Compute precession moment) • Momentum Control	<i>SunPresent(any eye lit)</i> • Eclipse logic
	SWAS	<i>Ac_USun_BF</i> • TRIAD • Kalman Filter Residuals	<i>Ac_USun_BF</i> • Digital Sun Control • Momentum Control	<i>SunPresent(any eye lit)</i> • Eclipse logic

- Mission/Client Matrices
 - Captures model usage by other models
 - Domain engineering product: background for modeling decisions

Method Evaluation: Observations

- Should make minimum necessary change to existing processes
 - Build on current FSB approaches
- BUT
 - Assure that reusable products are really reusable by intended audience
- Process and product are variables you manipulate to make
 - Application engineering easy for all users
 - Application building techniques easy to automate

Domain Assessment: Observations

- Assessment process must be
 - Repeated with each major iteration
 - Include payback assessment
- Criteria are needed for decomposition of domain into subdomains and models
- “Class” and “Model” are sometimes used (erroneously) as synonyms

Prototyping Observations (1 of 2)

- Variations in mission orbit carried forward to selection of models for an application
 - This is a first, simple, and successful step
 - Variabilities are more complex in other subdomains
- Prototyping to date hasn't made clear distinction between domain and application products
 - We were able to determine what was implicit in source code
 - To be usable, decision model and application engineering must be made explicit and traceable
- Products (so far) are developer oriented, other users include GNC analysts, testers and tool developers

Prototyping Observations (2 of 2)

- Products should be as consistent as possible
 - Easier application engineering
 - Less complex task to automate processes
- Design variations between prototypes
 - Whether generics/templates are used
 - Direct message passing or use of “switchboard” code
 - *Are these variations essential to the problem or not?*

Product consistency depends on architecture

Future work

- Complete process and product definitions
 - Consider different user perspectives
 - Domain and application engineers
 - Analysts, developers, and testers
- Complete and evaluate prototypes
 - Keep architecture as consistent as possible
 - Run small, realistic system on flight test bed
 - Use prototype to evaluate both process and architecture