Technical Debt Aggregation in Ecosystems

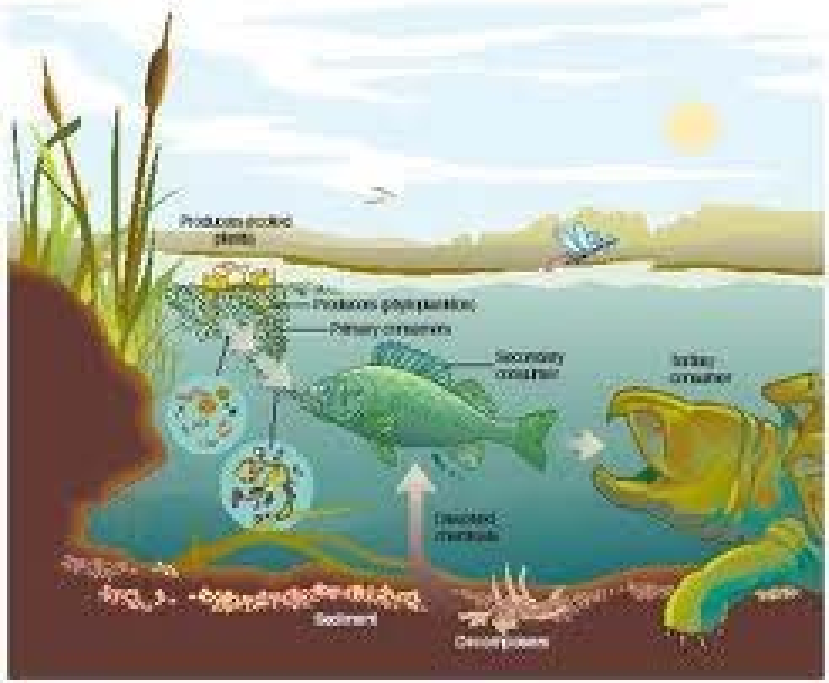
John D. McGregor, J. Yates Monteith, and Jie Zhang

Clemson University

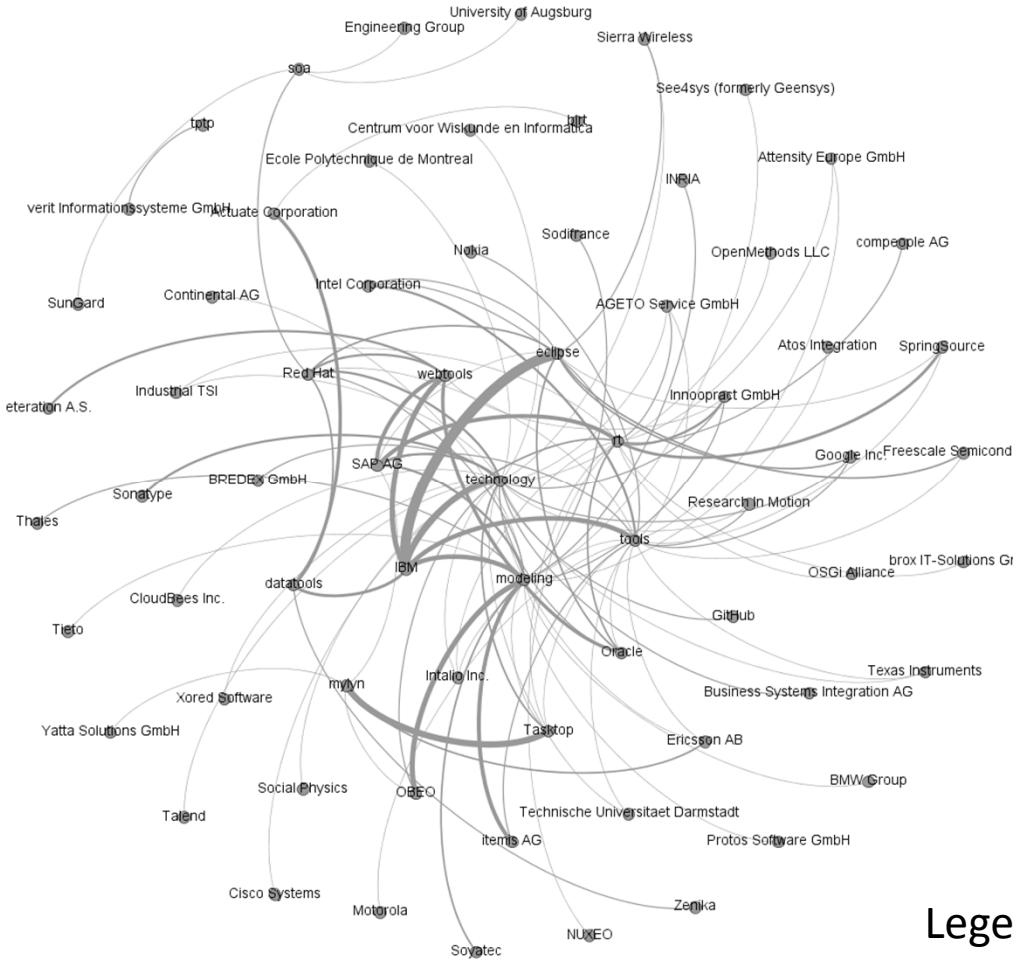
CLEMSON UNIVERSITY

Ecosystems

- An ecosystem is a context in which organizations produce and consume software, transact business with each other, and perform these activities in new and innovative ways
- Three primary types of ecosystems: business ecosystems, software ecosystems and innovation ecosystems
- Those three types of ecosystems provide different views on the entire ecosystem. Together they provide more complete information to the decision maker when the ecosystem involves a software product line and its organization.

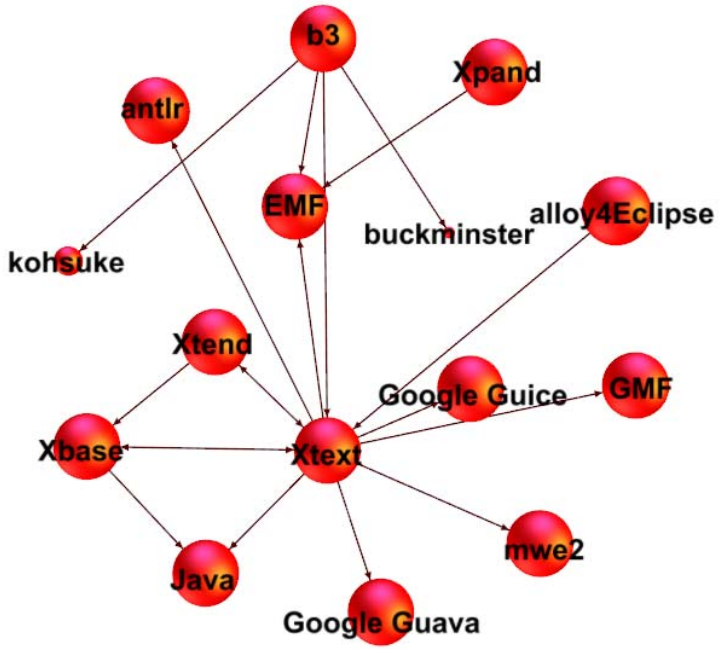


There are linkages between the business and software views



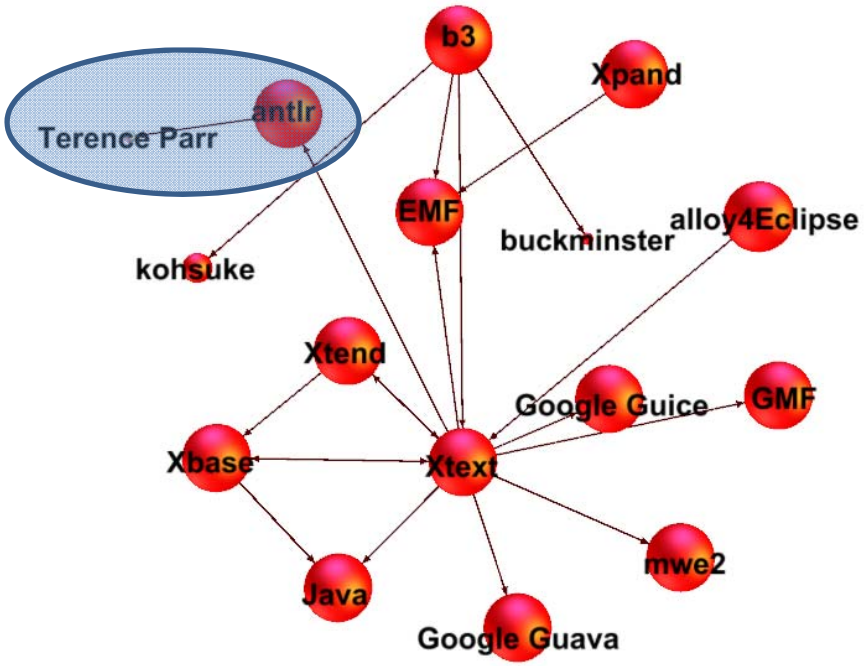
Legend: arrow runs from Organization to Eclipse top level project.

Xtext project from Eclipse – one level of supply chain



Strategic reuse sets up software supply chains that are prone to the same problems as hard goods supply chains.

Looking further we find a problem



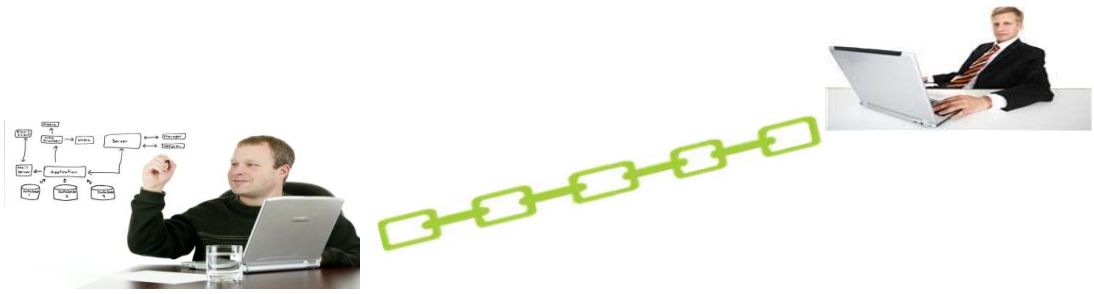
Looking further we find that antlr is copyrighted by one person. This is a definite risk in a software product supply chain.

Strategic reuse in an ecosystem

- Ecosystems are everywhere but flourish where strategic reuse occurs.
- Often there is a “platform” used as the basis for a set of products
- In a software product line the platform is the core asset base
- Strategic reuse propagates assets; whether they are good or bad

Propagation of debt in supply chains

- A supply chain in the software view represents a sequence of “uses” dependencies, linking software elements including compilation units, software libraries, frameworks and other units of software deployment.
- Our questions focus on how technical debt is propagated along those dependencies and how technical debt accumulates at various points in the chain.



Association of debt

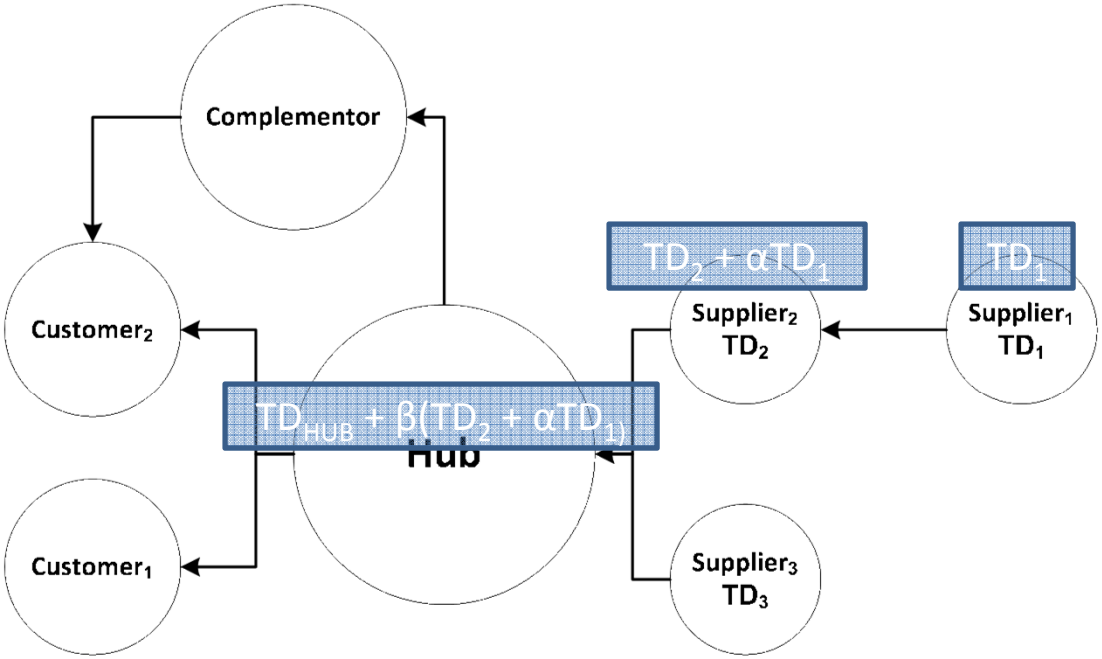
- Technical debt is associated with a particular software asset although it is due to a decision by a member of the organization that produces the asset.
- An organization can produce multiple assets each of which has a unique amount of technical debt associated with it.
- One question is whether, as an asset is delivered to other organizations, the technical debt flows with the asset or whether it is an attribute of the combination of asset and organization.

Consider a house mortgage
which is made to an individual
but secured by the asset.

We hypothesize that

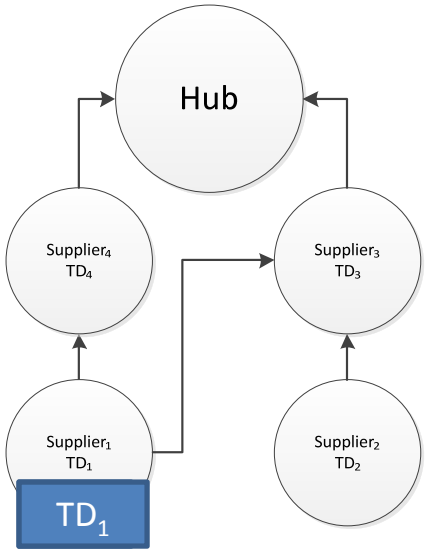
- The technical debt for a newly created asset is the sum of the technical debt incurred by the decisions during development of the asset and some amount based on the quality of the assets integrated into its implementation.
- The technical debt of an asset is not directly incurred by integrating an asset in object code form, but there is an indirect effect on the user of the asset.

Supply Chain 1



- The Hub in the ecosystem maintains the core asset base and acquires assets from suppliers.
- Some of the product development projects use the software assets as is and some build on it to provide something bigger.
- As the technical debt incurred in Supplier$_1$ flows to Supplier$_2$ and on does the impact of that debt change? Does it diminish as the asset from Supplier$_1$ gets more deeply encapsulated?

Supply Chain 2



- In this case there is a common supplier to two other suppliers.
- Does the core asset base in the Hub reflect the debt from $Supplier_1$ along both supply chains?

Scenario 1 – Direct impact

- We hypothesize that any user of an asset that utilizes the source code of an asset accepts some, if not all, of the technical debt carried by that asset.
- If $Supplier_2$ utilizes a source code based asset from $Supplier_1$, $Supplier_2$ may have to be intimately familiar with the code supplied by $Supplier_1$. For example, when using inheritance as a variation mechanism.
- Technical debt, from static attribute sources such as lack of adherence to naming conventions, incomplete or inadequate documentation and unmanaged complexity, affect such users by requiring more resources to make a specific modification. How much of the debt do they pay off in order to improve their own product?

Scenario 2 – Indirect impact

- We hypothesize that a user of the object code feels the effects of technical debt but does not incur any debt from the asset.
- The hub provides its product in object code format to a customer. The customer begins to use the product and experiences a random failure every few days. (The supplier decided to compress testing into fewer days than requested by the testing team.)
- The customer has not assumed any of the technical debt of the product but will feel its effects. Because of the technical debt in the hub's supply chain the hub will release new versions, which solve the failure problem, more slowly than would otherwise be the case.

	Source level	Object level
Static attribute	Comments in source	Memory size
Dynamic attribute	Extraneous code	Performance

Summary

- Our interest in technical debt stems from our work on ecosystem modeling and strategic reuse in a software product line.
- The quality of the core asset base is critical to the success of the product line organization.
- Technical debt may be a useful tool in measuring this quality.
- Our work continues: debt vs options
  - Debt is an obligation; option is a right but no obligation
  - How does variation (e.g., optional features) factor in?