Software Ecosystems
Overview and Implications

Jan Bosch
Overview – The Power of Three

- Define the notion of software ecosystems
- External software ecosystem study – learnings
- Implications for HOW we develop software
paradigm shift – an analogy

Anaximander
ca. 610 – 545 BC

Isaac Newton
1643-1727

Cicero
106 BC – 43 BC

Galileo Galilei
1564-1642

Albert Einstein
1879-1955

Charles Darwin
1809-1882

Thomas S. Kuhn
1922-1996
defining software ecosystems

A human ecosystem consists of actors, the connections between the actors, the activities by these actors and the transactions along the connections. For this discussion, we recognize commercial and social ecosystems.

A commercial ecosystem consists of actors, i.e. businesses, suppliers and customers, activities by these actors, goods and services and the transactions that take place over those connections. Transactions include financial transactions, but also information and knowledge sharing, inquiries, pre- and post-sales contacts, etc.

A software ecosystem provides the set of solutions that enable, support and automate the activities and transactions by the actors in the associated social or business ecosystem and the organizations that provide these solutions.

A software ecosystem consists of a software platform, a set of internal and external developers, a community of domain experts and a community of users that compose relevant solution elements to satisfy their needs.
defining software ecosystems

A human ecosystem consists of actors, the connections between these actors, the activities by these actors and the transactions along the connections. For this discussion, we recognize commercial and social ecosystems.

A commercial ecosystem consists of actors, i.e. businesses, suppliers and customers, the connections between these actors, the activities by these actors, goods and services and the transactions that take place over these connections. Transactions include financial transactions, but also information and knowledge sharing, inquiries, pre- and post-sales contacts, etc.

A software ecosystem consists of a software platform, a set of internal and external developers, a community of domain experts and a community of users that compose relevant solution elements to satisfy their needs.
### Software Ecosystems Taxonomy

<table>
<thead>
<tr>
<th>Category</th>
<th>Platform</th>
<th>Focus for Intuit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>End-user programming</strong></td>
<td>desktop</td>
<td>MS Excel, Mathematica, VHDL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yahoo! Pipes, Microsoft PopFly, Google’s mashup editor</td>
</tr>
<tr>
<td></td>
<td>web</td>
<td>none so far</td>
</tr>
<tr>
<td></td>
<td>mobile</td>
<td>none so far</td>
</tr>
<tr>
<td><strong>Application</strong></td>
<td>desktop</td>
<td>MS Office, SAP*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SalesForce, eBay, Amazon, Ning</td>
</tr>
<tr>
<td></td>
<td>web</td>
<td>none so far</td>
</tr>
<tr>
<td></td>
<td>mobile</td>
<td>none so far</td>
</tr>
<tr>
<td><strong>Operating system</strong></td>
<td>desktop</td>
<td>MS Windows, Linux</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Google AppEngine, Yahoo developer, Coghead, Bungee Labs</td>
</tr>
<tr>
<td></td>
<td>web</td>
<td>Nokia S60, Palm, Android, iPhone</td>
</tr>
<tr>
<td></td>
<td>mobile</td>
<td></td>
</tr>
</tbody>
</table>
one view of the Intuit ecosystem

World of opportunities: jobs to be supported and automated

- Compositional applications, e.g. accounting, customer, employee, payments, etc.
- Domain functionality, e.g. accounting, customer, employee, payments, etc.
- Non-differentiating, generic functionality, e.g. subscription, billing, entitlement, etc.

Experiment & innovate

- Platformize
- Commoditize
- Maintain sustainable competitive advantage

UX, workflow & data composition

Ecosystem platform

Breadth of applicability

Value to individual user
why software ecosystems

- Increases value of the core offering to existing users
- Increase attractiveness for new users
- Increase stickiness of the application platform
- Accelerate innovation through open innovation in the ecosystem
  - Collaborate with partners in the ecosystems to share cost of innovation
  - Platformize functionality developed by partners in the ecosystem (once success has been proven)
- Decrease TCO for commoditizing functionality by sharing the maintenance with ecosystem partners
study of external software ecosystems

How do we manage variability & configurability for customers?

How do we maintain consistent user experience between Intuit and ecosystem joblets?

How do we maintain dynamic composition of joblets by customers?

Do ecosystem joblets have the same access to domain services and data as Intuit joblets?

What mechanisms exist to insert ecosystem domain services into other domain service workflows?

Does Intuit provide a development environment for ecosystem developers?

Does Intuit provide access to data defined by ecosystem developers to other ecosystem developers?

Can ecosystem developers host their solutions outside Intuit’s hosting infrastructure?

Are other platforms allowed in the ecosystem and, if so, are these integrated?

Can ecosystem developers store data outside Intuit’s repository?

Can ecosystem developers access aggregated customer (DAAA) data?

Do we charge developers for developing and hosting in our ecosystem (beyond revenue share)?

Can ecosystem developers host their solutions outside Intuit’s hosting infrastructure?

Are other platforms allowed in the ecosystem and, if so, are these integrated?

Can ecosystem developers store data outside Intuit’s repository?

Can ecosystem developers access aggregated customer (DAAA) data?
learnings from external software ecosystems study

Customers first; developers second
Our QuickBooks, Quicken, TurboTax and ProTax franchises are our tickets to entry
A seamless desktop / web / mobile integration will be crucial for at least a decade
Customers adoption of our core web services (offerings) is critical for developer involvement

Platform should be in the middle of every transaction
Minimize interaction outside (bypassing) Intuit platform
Allow for storing application specific data inside Intuit hosting platform
Host solutions inside Intuit’s hosting platform
Offer a set of “connecting services” that take the burden from developers
Facilitate inclusion into the workflow to minimize the need to replicate the workflow outside the Intuit platform
**Learnings from external software ecosystems study**

Proactively incorporate “horizontalizing” / commoditizing functionality and data models into the platform:
- The platform needs to consist of small business domain data models, functionality and user interface solutions.
- At first signs of consolidation appear, incorporate 3rd party functionality (and possibly one of the companies) into the platform.
- Drive commoditized components into open-source or replace with COTS.

Communicate clear, multi-year roadmaps to avoid unintentional heads-on composition between platform and external developers:
- Publish an updated roadmap concerning data models, domain services, user experience solutions and compositionality with each release of the platform; guarantee that external communication is aligned with internal knowledge.

Model platform as the next computing platform abstraction layer:
- Use our leading position on the desktop to claim similar leadership on the web – we are providing *the* platform for small businesses (not *a* platform).
- We need to move fast, partner where appropriate and establish our “shaping view” for our ecosystem.
from integration-centric software engineering ...

manual integration  
global R&D  
pre-integrated products  
large R&D teams  
roadmapping & req. mgmt  
build & maintain  
yearly cycles  
meetings  
resource allocation
from integration-centric software engineering ...

software product lines
global software development
software ecosystems

causing

unacceptable complexity and coordination cost
... towards *composition*

- teams releases frequently, when they want
- teams are self-selected (2 pizza rule)
- teams are self-directed (little roadmapping)
- architecture prioritizes simplicity (3 API rule)
- customers compose their own products
- teams can be external (ecosystem)
- components are backward compatible and negotiate interfaces
- architectural compositionality
implications for how we build software

- From **process** to **ARCHITECTURE**
- From **centralized** to **DECENTRALIZED**
- From **planning** to **EXPERIMENTATION**
- From **long cycles** to **SHORT CYCLES**
- From **large teams** to **SMALL TEAMS**
- From **internal** to **ECOSYSTEM**
- From **CMM(I)** to **AGILE**
- From **cathedral** to **BAZAAR**
Not your job?
define the notion of software ecosystems
external software ecosystem study - learnings
implications for HOW we develop software