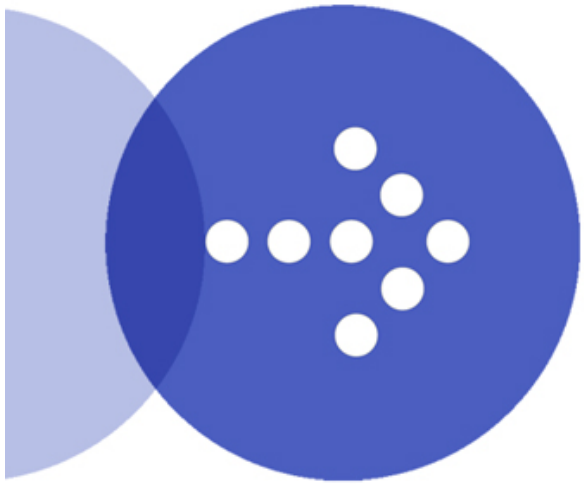
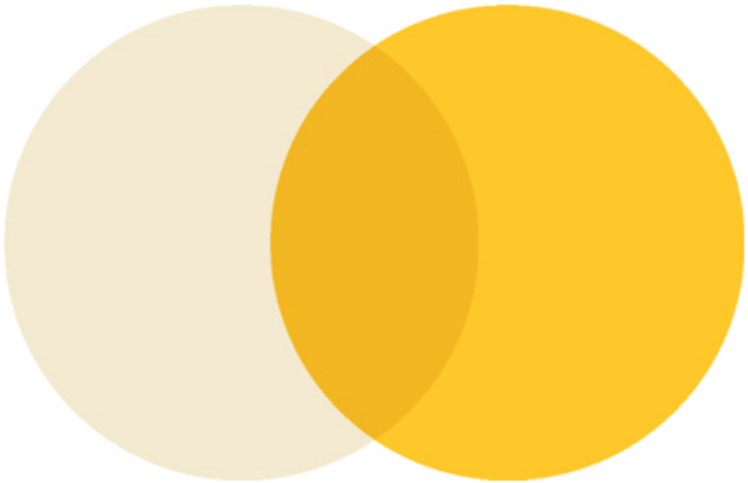
Towards Compositional Software Engineering

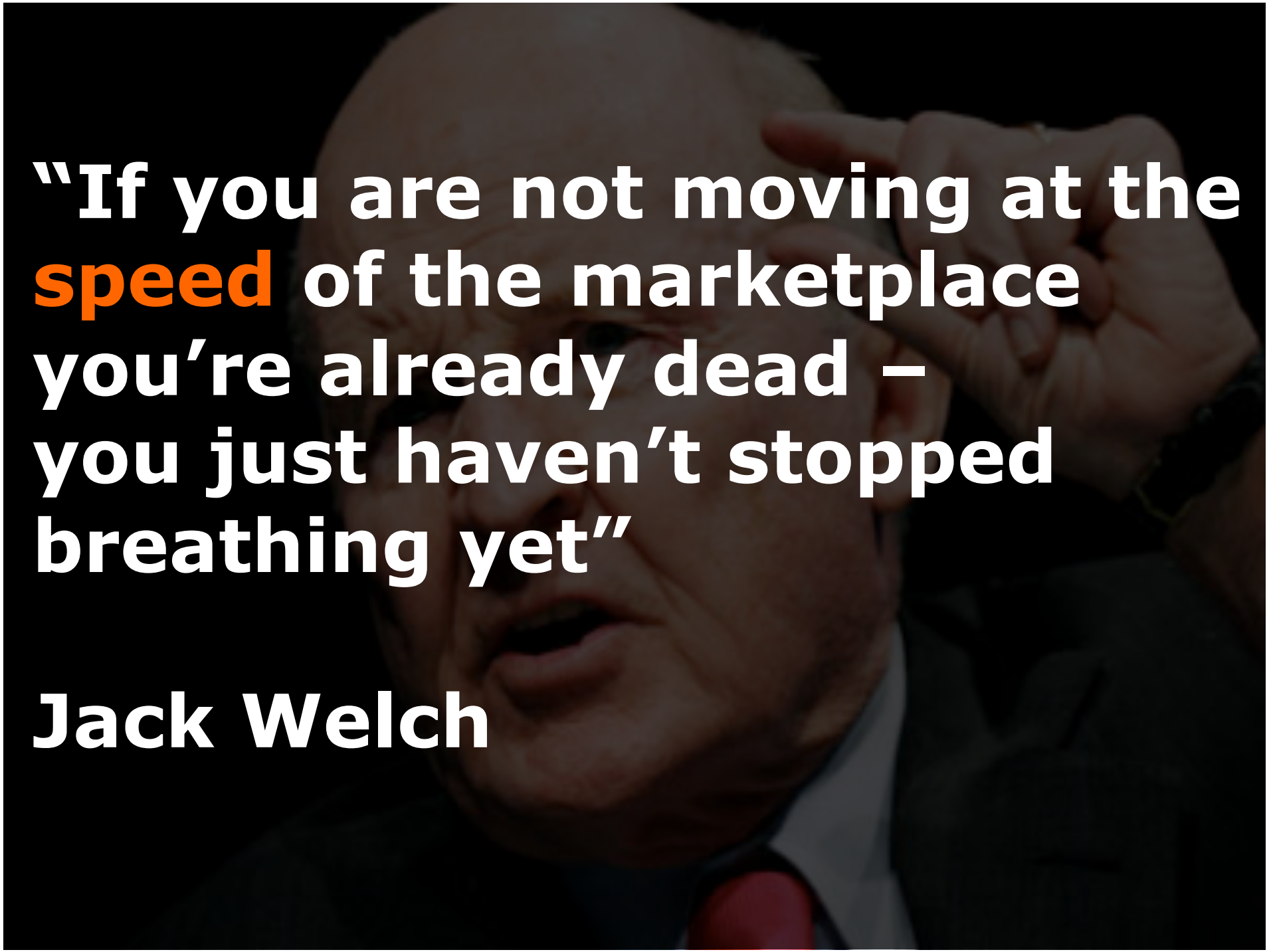
Jan Bosch
VP, Engineering Process
Professor of Software Engineering

September 22nd, 2010





intuit


"If you are not moving at the
speed of the marketplace
you're already dead –
you just haven't stopped
breathing yet"
Jack Welch

Three Key Take-Aways

- Increasing **SPEED** trumps ANY other improvement R&D can provide to the company – it is the foundation for everything else
- Software engineering is at an **inflection point** – from "integration-oriented" to "composition-oriented" software engineering
- In a world of **continuous deployment** and **software ecosystems**, automation is fundamental

Overview

- Vem är jag? Wie ben ik? Who am I?
- Introducing Intuit
- Speed matters: implications for software engineering
- Building *delightful* products
- Software ecosystems
- Implications for software engineering automation
- Conclusion

From Research to Industry

?

Engineering Process
(Intuit, USA)

Head of research lab
(Nokia, Finland)

Professor of software engineering
(RuG, Netherlands)
(BIT, Sweden)

Industrial development

Industrial research

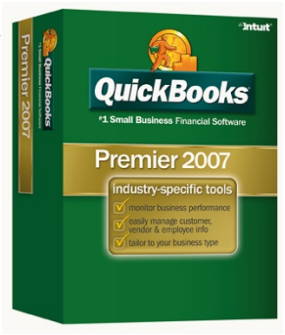
Academia (+ consulting)

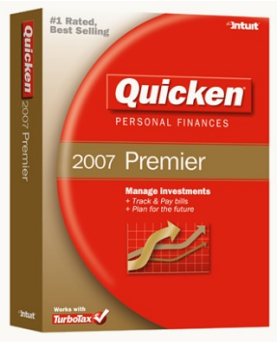
intuit

Intuit Company Information

Who We Are…

A leading provider of business and financial management solutions





- Founded in 1983
- FY 2010 revenue of $3.5 billion
- Intuit is traded on the NASDAQ: INTU
- Employs around ~8,000 people
- Major offices across the U.S. and in Canada and the United Kingdom
- More than 40 million people use our QuickBooks, Payroll, Payments, TurboTax, Digital Insight and Quicken products and services.

intuit

Mission: why we exist as a company…

To be a premier innovative growth company that improves our customers' financial lives so profoundly… they can't imagine going back to the old way

We serve these end customers

- Consumers
- Small Businesses

…and those who serve them

- Accountants
- Financial Institutions
- Health Care Players

"Better Money Outcomes"

Financial… making & saving money, grow & profit

Productivity… turning drudgery into time for what matters most

Compliance… without even having to think about it

Confidence… from the wisdom & experience of others

intuit

Proven formula: lots of delighted customers…

Help families find $1,000 annually… $400M in consumer savings

INTUIT
Improving
40M
Lives

Help small businesses be 20% more profitable… Customers revenues ~20% of U.S. GDP, pay 1 in 12 American workers

Help people get the maximum tax refund… $33B in tax refunds, 1 out of every 3 tax returns e-filed

Improve FI profit per customer by 20%… IB customers equal to the 5th largest U.S. bank

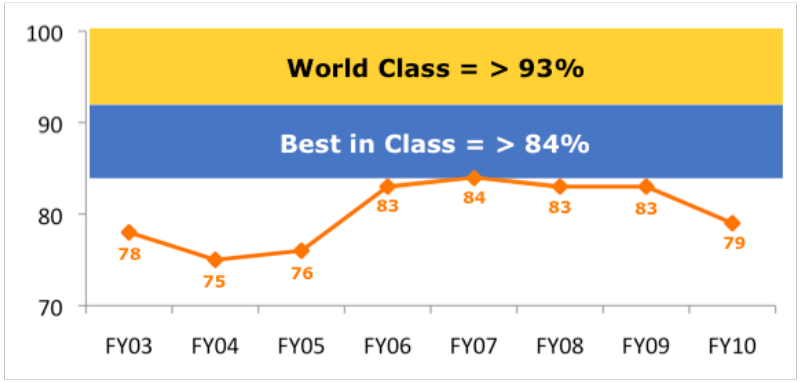
Help accountants be 20% more productive today… Serve half of all accounting firms

INTUIT

Proven Formula: talented & engaged employees

Most Admired: Software Industry

Strong Employee Engagement



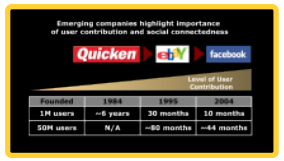
Fortune Top 100 Places to Work

intuit

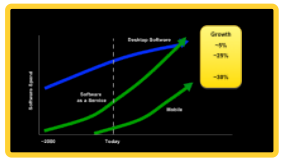
Secular Shifts: transforming our company…

Trends

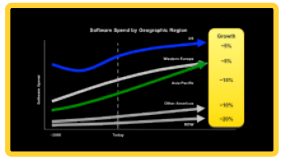
Demographic Shifts



Value Creation Shifts



Technology Shifts



Geographic Shifts

Implications

Intuit is driving:

"Connected Services"

- **Software-Advantaged Services**
- **Software-as-a-Service**
- **Platform-as-a-Service**

Intuit is embracing:

Social

capitalize on our large and growing customer bases to unleash the collective power of user contributions, behaviors and data

Mobile

deliver "in the pocket" when that is the preferred solution

Global

employ the world's talents to find & solve important problems around the globe

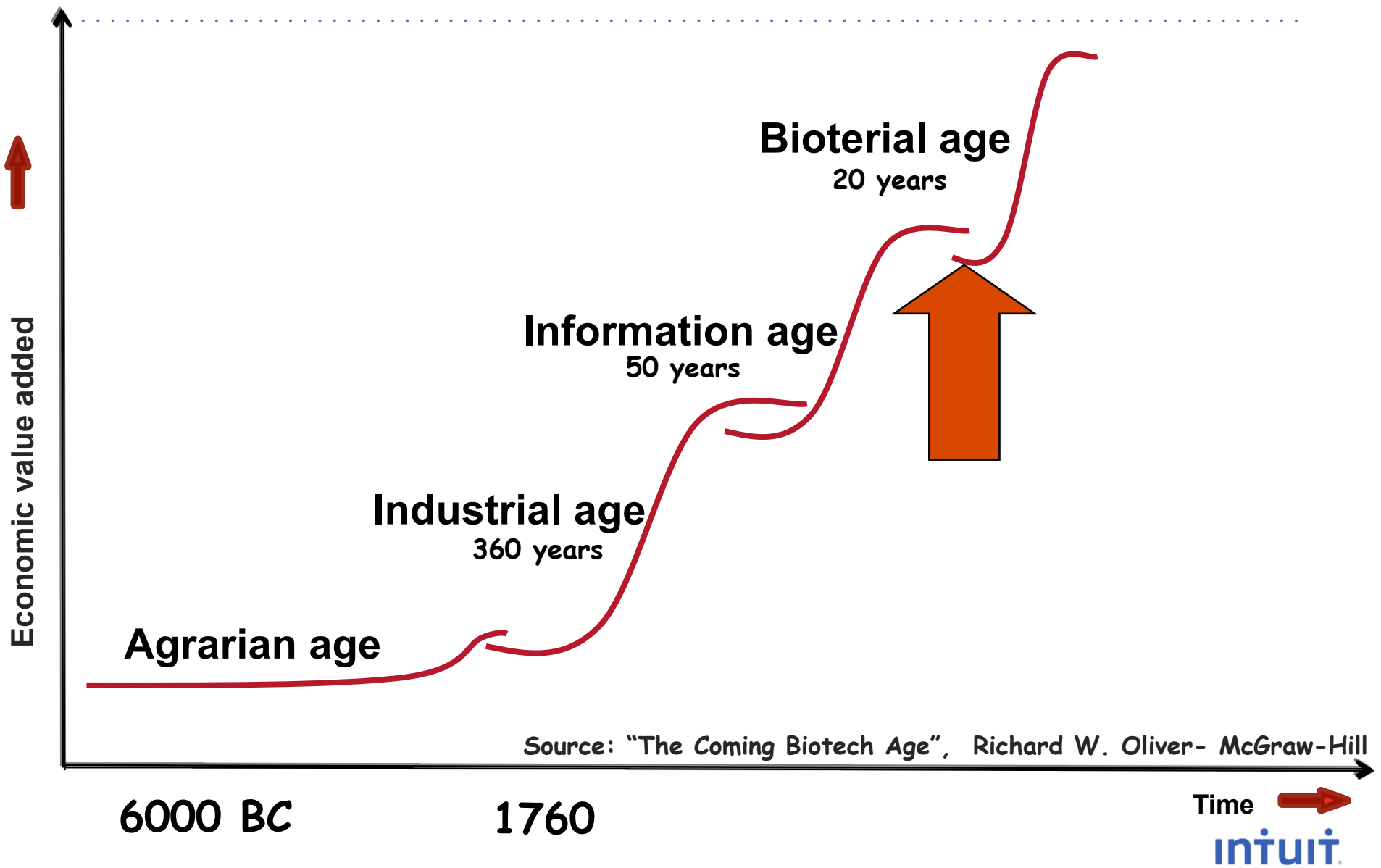
intuit

Overview

- Vem är jag? Wie ben ik? Who am I?
- Introducing Intuit
- Speed matters: implications for software engineering
- Building *delightful* products
- Software ecosystems
- Implications for software engineering automation
- Conclusion

intuit

Where are we going? How fast?



intuit

Accelerating User Adoption

Value Creation Shifts

Emerging companies highlight importance of user contribution and social connectedness

Founded	1984	1995	2004
1M users	~6 years	30 months	10 months
50M users	N/A	~80 months	~44 months

intuit

Need for Speed in R&D – An Example

- Company X: R&D is **10%** of revenue, e.g. 100M$ for a 1B$ product
- New product development cycle: **12 months**

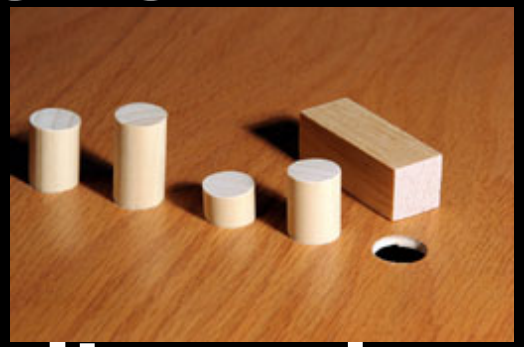
- Alternative 1: improve efficiency of development with 10%
  - **10 M$** reduction in development cost
- Alternative 2: reduce development cycle with 10%
  - **100M$** add to top line revenue (product starts to sell 1.2 months earlier)

No efficiency improvement will outperform cycle time reduction

Integration-centric software engineering

software product lines
global software development
software ecosystems

causing



unacceptable complexity and coordination cost

Web 2.0 Rules to SW Development (1/2)

Focus on Simplicity

Team size

- 3x3 = 3 persons x 3 months (Google)
- 2 pizza rule (Amazon)
- Principle: What is required is a team, where the roles are defined and each member has the right skill for that role, and following a lean, agile, method — all focused on the customer.

Release cycle

- Weeks, not months
- Continuous deployment
- Principle: short cycles are key for agility, speed and decoupling

Architecture

- 3 API rule
- Mash-ups and web services
- Principle: architecture provides simplicity, compositionality and is designed in parallel with software development

Focus on one thing: Minimize Dependencies

Web 2.0 Rules to SW Development (2/2)

Focus on Simplicity

Requirements and Roadmapping

- Each team (3 persons) announces wha… to release
- Some (QA) requirements are shar… rd, e.g. performance, latency, etc.
- Principle: the cost of ove… much lower than the cost of synchroni… maps and plans

Process

- CMMi and o… rity approaches address the sympto… ause
- Co… ensive illusion causing LOTS of in… e system
- Prin… rchitecture not process should manage coord… ation and alignment

The POWER of Public Humiliation

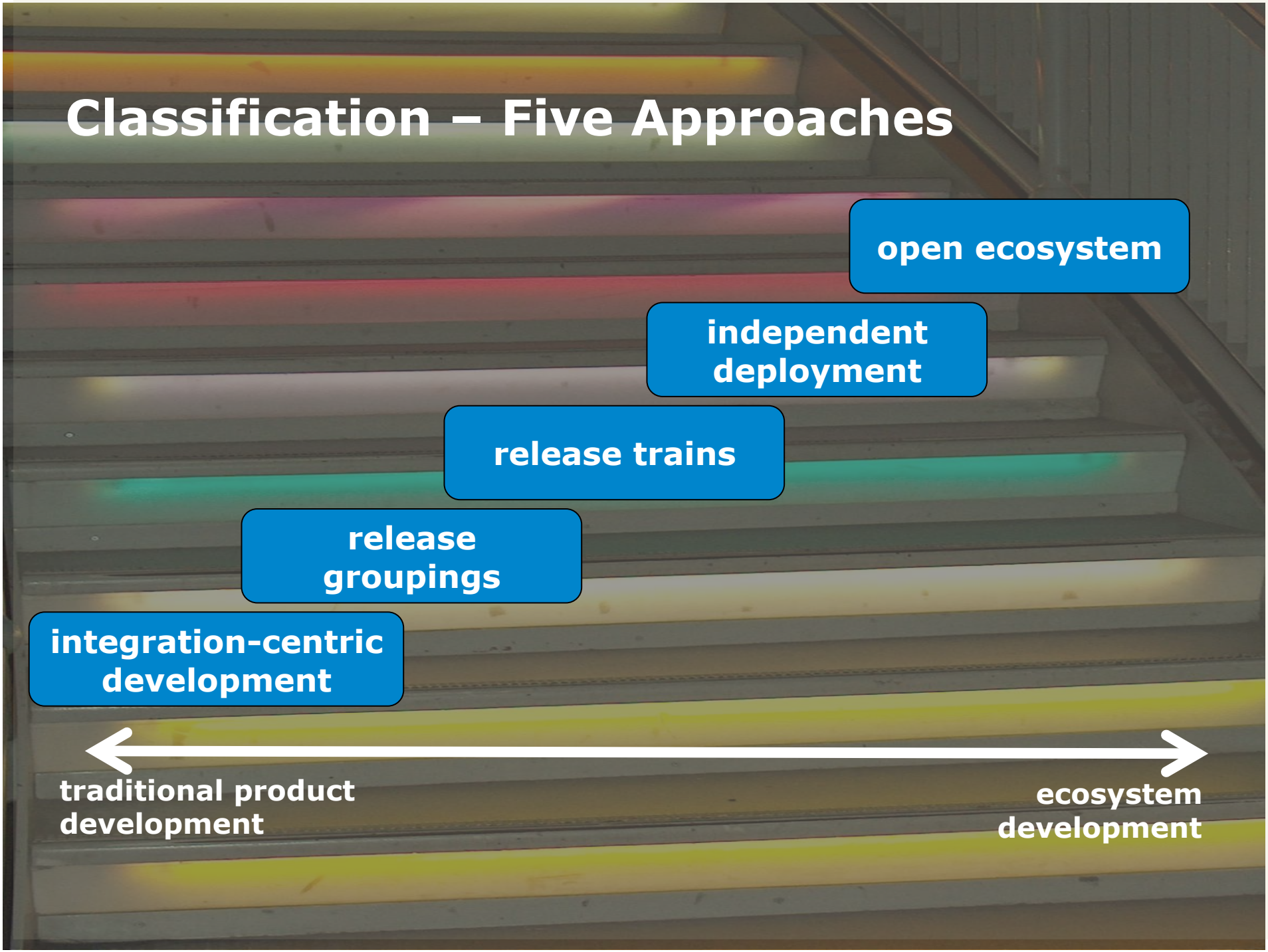
From the Cathedral to the Bazaar

Towards Composition …

Implications for Software Engineering

- 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**

Classification – Five Approaches

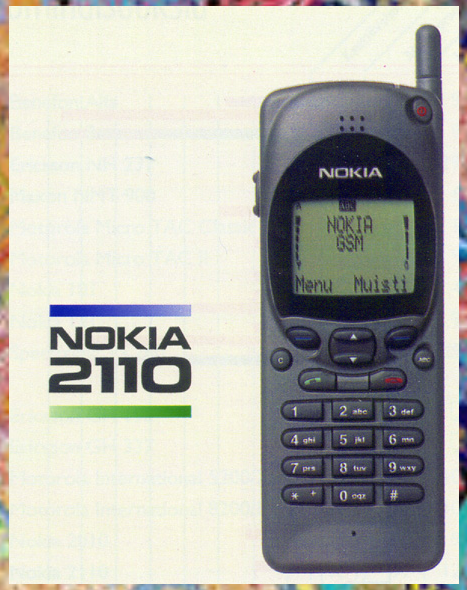


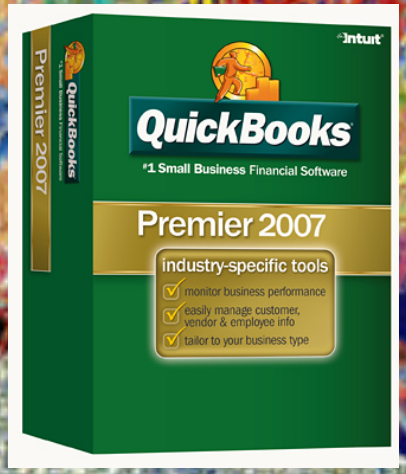
Overview

- Vem är jag? Wie ben ik? Who am I?
- Introducing Intuit
- Speed matters: implications for software engineering
- Building *delightful* products
- Software ecosystems
- Implications for software engineering automation
- Conclusion

intuit

What Do These Product Have in Common?





Designing Pleasurable Products

Hierarchy of Consumer needs

"People seek pleasure"

Jordan's four pleasures framework (based on Tiger 1992):

Physio-pleasure
- Pleasure from sensory organs, e.g. tactile feedback

Socio-pleasure
- Enjoyment from social interactions

Psycho-pleasure
- Cognitive and emotional responses, e.g. usability

Ideo-pleasure
- Supporting people's values, e.g. green values

Jordan, P (2002): Designing Pleasurable Products, Taylor and Francis.

"Design for Delight" at Intuit

= WOW

Going beyond customer expectations in delivering ease and benefit, evoking positive emotion throughout the customer journey…
…So folks buy more & tell their friends

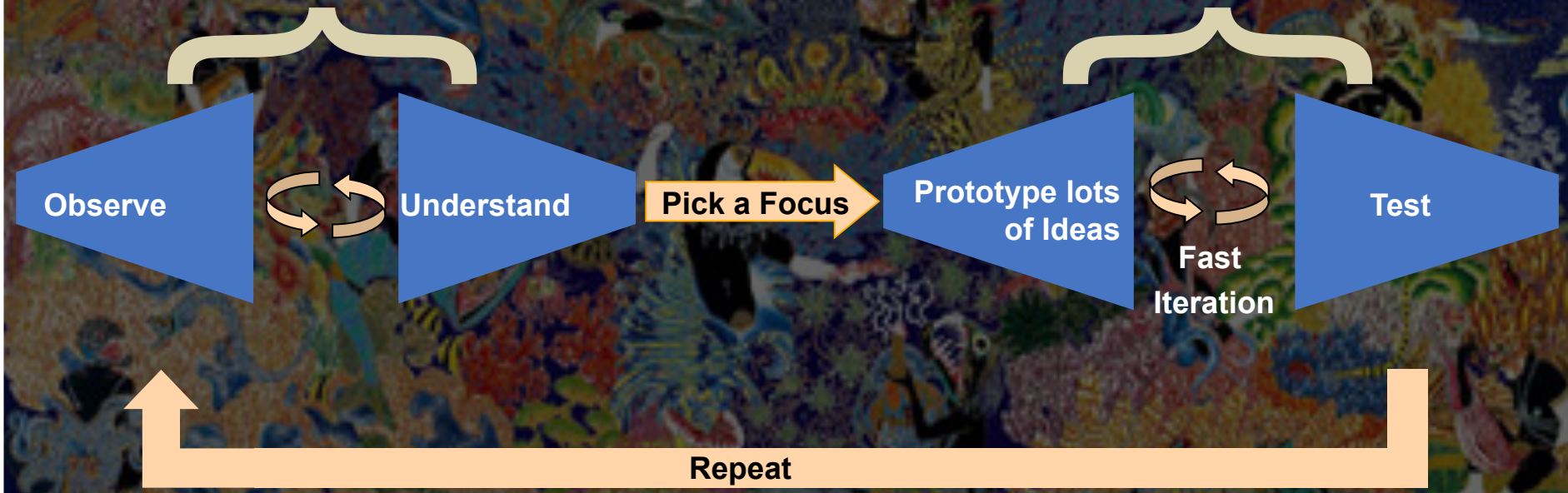
Benefit = the improvement in the customer's life or business outcome

Growing our business is the goal

The "How"

Uncover what's most important to customers

In that focus, create better solutions, within available resources



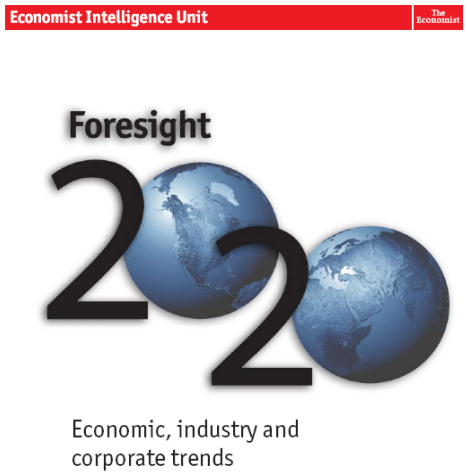
Intuit Design4Delight Framework

Overview

- Vem är jag? Wie ben ik? Who am I?
- Introducing Intuit
- Speed matters: implications for software engineering
- Building *delightful* products
- Software ecosystems
- Implications for software engineering automation
- Conclusion

Towards Web 3.0

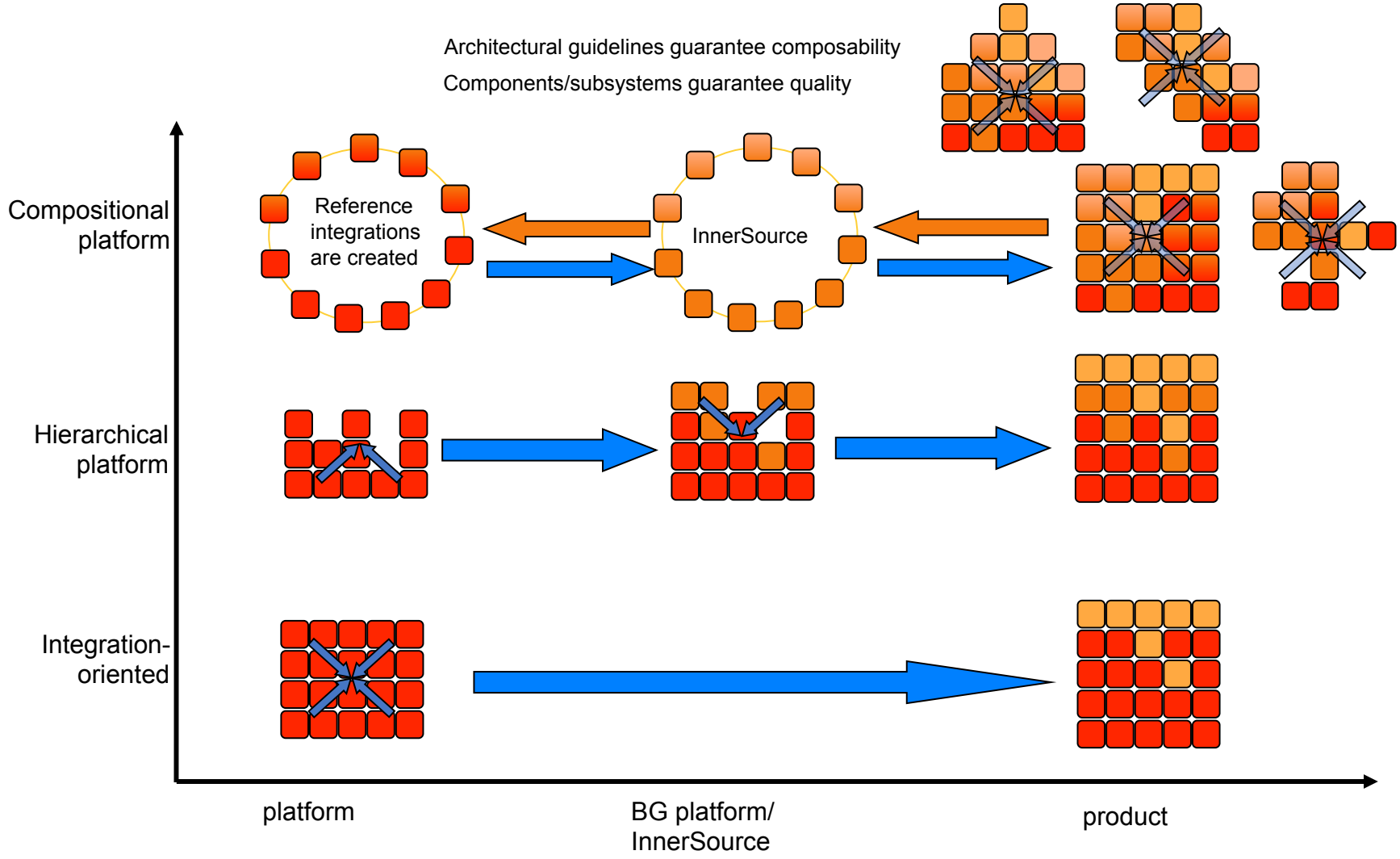
3 Atomisation. Globalisation and networking technologies will enable firms to use the world as their supply base for talent and materials. Processes, firms, customers and supply chains will fragment as companies expand overseas, as work flows to where it is best done and as information digitises. As a result, effective collaboration will become more important. The boundaries between different functions, organisations and even industries will blur. Data formats and technologies will standardise.



"My prediction would be that Web 3.0 will ultimately been seen as applications which are pieced together. There are a number of characteristics: the applications are relatively small, the data is in the cloud, the applications can run on any device, PC or mobile phone, the applications are very fast and they're very customizable. Furthermore, the applications are distributed virally: literally by social networks, by email. You won't go to the store and purchase them... That's a very different application model than we've ever seen in computing." —Eric Schmidt

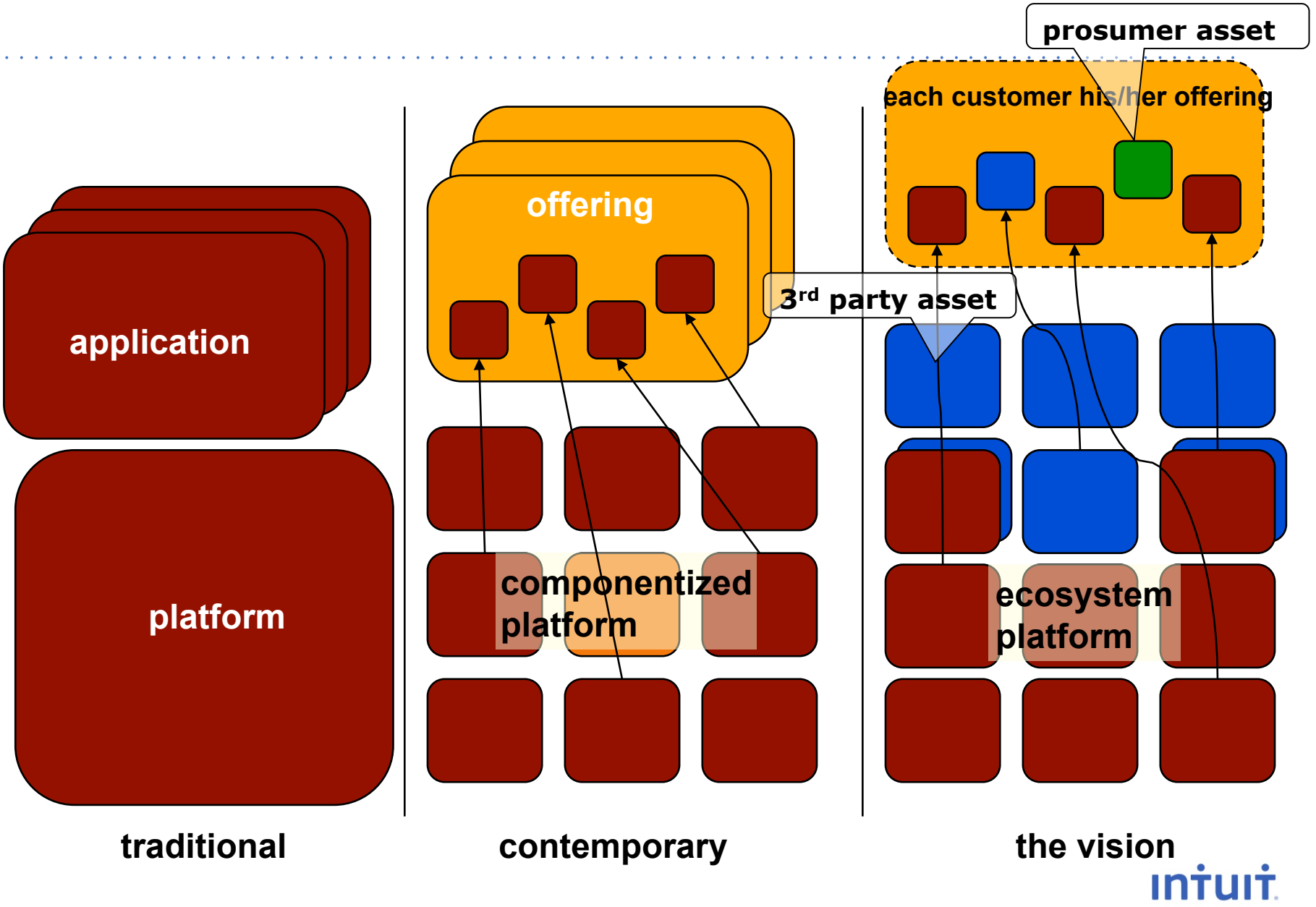
intuit

Toward Product Composition …



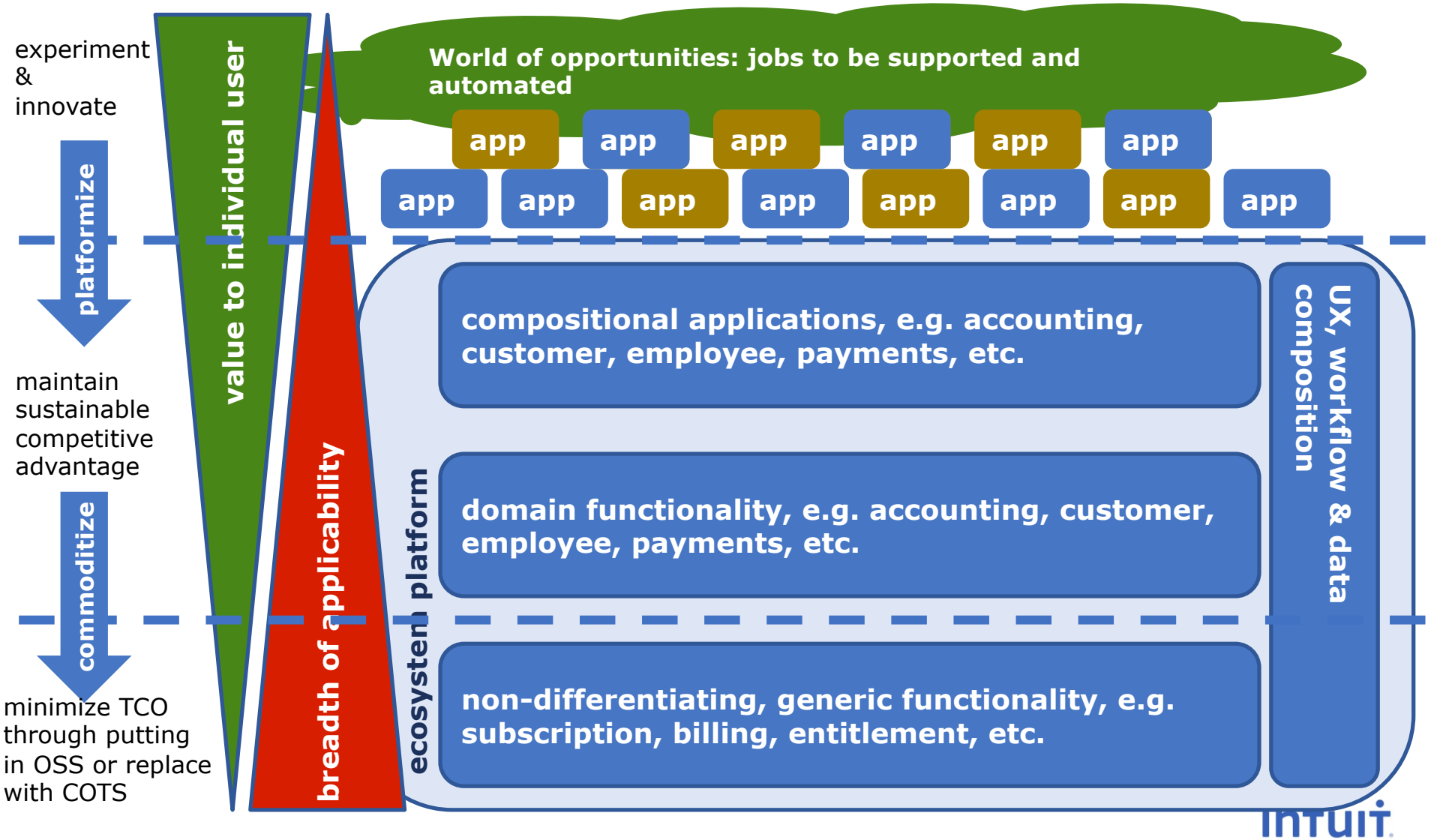
intuit

From Pre-Packaged Offerings to Customer-Assembled



intuit

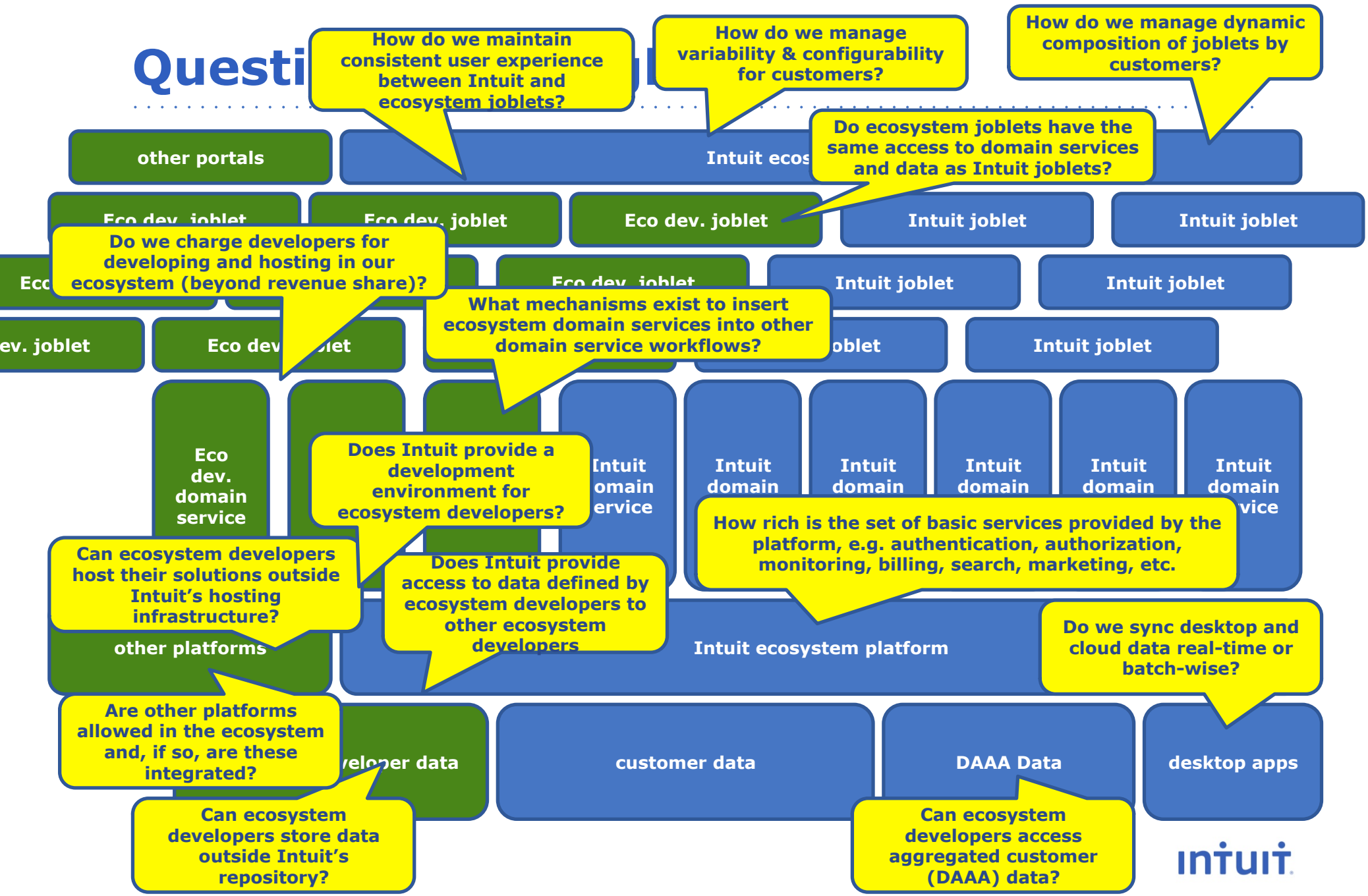
One View of the Intuit Ecosystem



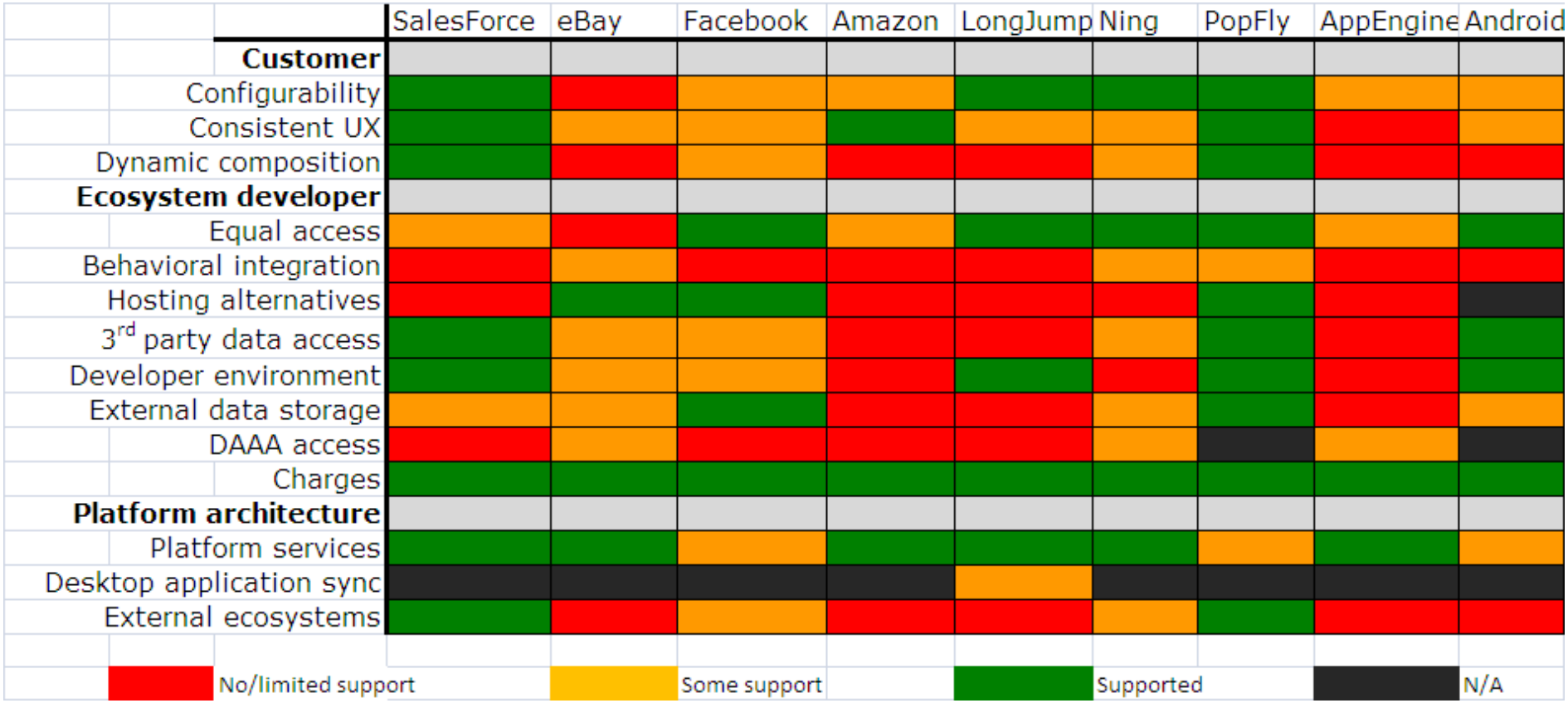
intuit

Classifying Software Ecosystems

category / platform	desktop	web	mobile
end-user programming	MS Excel, Mathematica, VHDL	Yahoo! Pipes, Microsoft PopFly, Google's mashup editor	*none so far*
application	MS Office	SalesForce, eBay, Amazon, Ning	*none so far*
operating system	MS Windows, Linux, Apple OS X	Google AppEngine, Yahoo developer, Coghead, Bungee Labs	Nokia S60, Palm, Android, iPhone


Questi
How do we maintain consistent user experience between Intuit and ecosystem joblets?
How do we manage variability & configurability for customers?
How do we manage dynamic composition of joblets by customers?
Do ecosystem joblets have the same access to domain services and data as Intuit joblets?
other portals
Intuit ecos
Eco dev. joblet
Eco dev. joblet
Eco dev. joblet
Intuit joblet
Intuit joblet
Do we charge developers for developing and hosting in our ecosystem (beyond revenue share)?
What mechanisms exist to insert ecosystem domain services into other domain service workflows?
Intuit joblet
Intuit joblet
Intuit joblet
Eco dev. domain service
Does Intuit provide a development environment for ecosystem developers?
Intuit domain service
Intuit domain service
Intuit domain service
Intuit domain service
Intuit domain service
Intuit domain service
How rich is the set of basic services provided by the platform, e.g. authentication, authorization, monitoring, billing, search, marketing, etc.
Can ecosystem developers host their solutions outside Intuit's hosting infrastructure?
Does Intuit provide access to data defined by ecosystem developers to other ecosystem developers
other platforms
Intuit ecosystem platform
Do we sync desktop and cloud data real-time or batch-wise?
Are other platforms allowed in the ecosystem and, if so, are these integrated?
customer data
DAAA Data
desktop apps
Can ecosystem developers store data outside Intuit's repository?
Can ecosystem developers access aggregated customer (DAAA) data?
intuit

Comparing Existing Ecosystems



intuit

Overview

- Vem är jag? Wie ben ik? Who am I?
- Introducing Intuit
- Speed matters: implications for software engineering
- Building *delightful* products
- Software ecosystems
- Implications for software engineering automation
- Conclusion

Implications

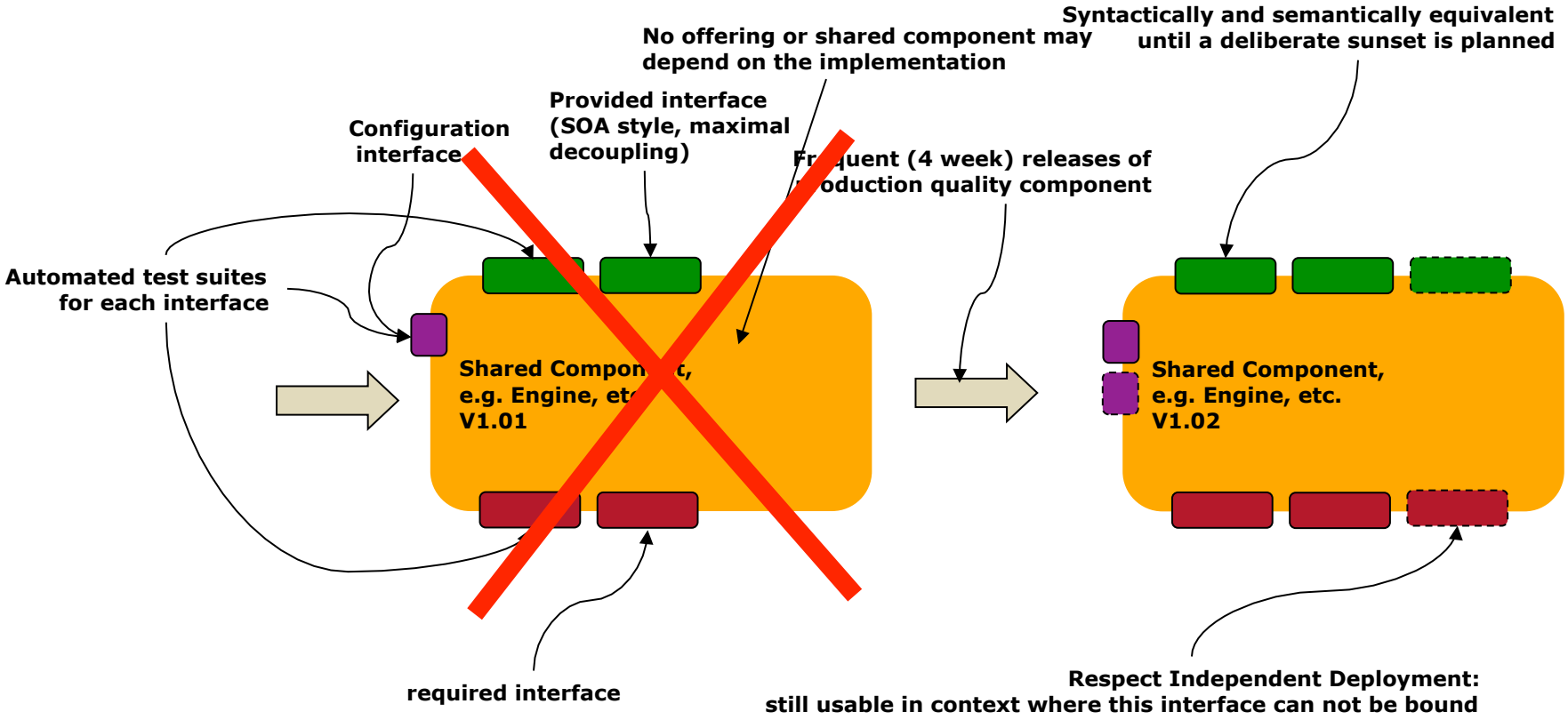
- Simplify, simplify, simplify
  - Make it easy to do the right thing, e.g. no versioning
  - Decouple components, teams and organizations
- Continuous Deployment
- Scale: Make teams effective in large systems
- Support software ecosystems
- Help manage design erosion

Simplify, Simplify, Simplify

Our life is frittered away by detail. Simplify, simplify, simplify! I say, let your affairs be as two or three, and not a hundred or a thousand; instead of a million count half a dozen, and keep your accounts on your thumb-nail – Henry Thoreau (Walden)

- Each design decision adds design rules and constraints that need to be observed by engineers
- Collectively, these decisions cause major complexity that decrease productivity
- **What to do**: Hide it, platformize it, make it happen automatically

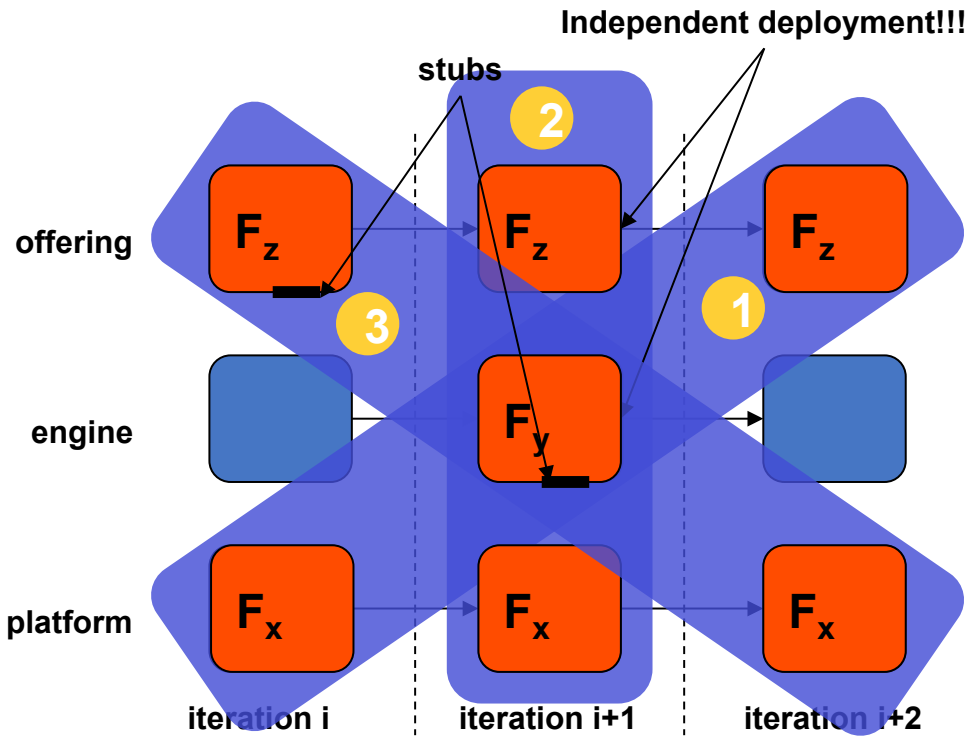
Decoupling: No Versions!



intuit

Decouple Components and Teams

1. Sequential feature development (90%)
2. Concurrent development, independent deployment enforced (8%)
3. Exploratory development (2%)



intuit

Strive For Continuous Deployment

- Software engineer checks in code => system compiles, links, tests and deploys the new code
- The automated QA infrastructure, NOT the engineer, is responsible for making sure the system does not go down
- If that's too much, aim for Independent Deployment
- If that's too much, aim for Release Trains

Open research topic: tool support for continuous deployment in safety/business critical systems

Scale: Make teams effective in large systems

- Components, especially over time, build major dependencies, complicating development
- Component teams are dependent on each other along the lines of the component dependencies
- Feature teams tend to make changes in all components affected by the feature, adding risk and affecting release schedules
- Systems that require software assets from multiple organizations suffer even more from dependencies between components

Open research topic: tool support for feature teams that mitigate aforementioned risks

Support Software Ecosystems

- Software ecosystems do not allow for process-based coordination
- Architecture forms the basis for coordination
- Enforced process by platform owner major source of frustration for external developers

Open research topic: Tool support to minimize/remove "process-based" interaction

Evolve architecture; fight erosion

Overview

- Vem är jag? Wie ben ik? Who am I?
- Introducing Intuit
- Speed matters: implications for software engineering
- Building *delightful* products
- Software ecosystems
- Implications for software engineering automation
- Conclusion

Speed

Increasing **SPEED** trumps ANY other improvement R&D can provide to the company – it is the foundation for everything else

- As a process, methods or tools professional, there is only ONE measure that justifies your existence: how have you helped teams move faster?
- Don’t optimize efficiency, optimize speed

Inflection Point

- Software engineering is at an **inflection point** – from "integration-oriented" to "composition-oriented" software engineering
- Design for automated compositionality, not manual integration
- Minimize dependencies
- Focus on small teams of engineers, give them direction and get out of their way

Automated Software Engineering 2.0

- In a world of **continuous deployment** and **software ecosystems**, automation is **fundamental**

- Simplify, simplify, simplify
- Continuous Deployment
- Scale: Make teams effective in large systems
- Support software ecosystems
- Help manage design erosion

Not My Job?!

Strong LEADERSHIP needed from YOU

intuit

Software Engineering Research

What it is NOT

- Science, i.e. proving that something can be done
- Research is leading
- Validation can be done "in-vitro"
- Researchers understand what is important
- Research can focus on one narrow, often technological, aspect
- A niche activity

What it is

- Engineering, i.e. establish the benefit of a solution
- Industrial practice is ahead of research
- Validation occurs "in-vivo"
- Without industrial experience, relevant research is hard
- Research needs to be holistic, addressing organizational, process and business issues
- Supporting a multi-billion dollar industry and critical

THANK YOU

Mount Shasta (CA) - 4,322m, July 2009