Rapid Software Architecture Exploration

Michael Keeling
@michaelkeeling

Conventional Wisdom:

It takes a long time to design software architecture.

Some Examples…

Quality Attributes Workshop (QAW)	2-3 Weeks
Architecture Trade-off Analysis Method (ATAM)	2-3 Weeks
Software Risk Evaluation Workshop	5 days
Empty Software Architecture Description Template (SEI)	30 pages

This is a problem for
Agile software architects.

One week is an eternity in Agile time.

Your Agile Project

How old your project feels (in “agile years”)

Same Age

Better Conventional Wisdom:

It takes a long time to design software architecture…

But you don’t need to design the whole system up front.

Agenda

- Theory
  - A Strategy for Architecture Exploration
- Practice
  - Rapid Exploration Practices
- Workshop!
- Reflection and Discussion

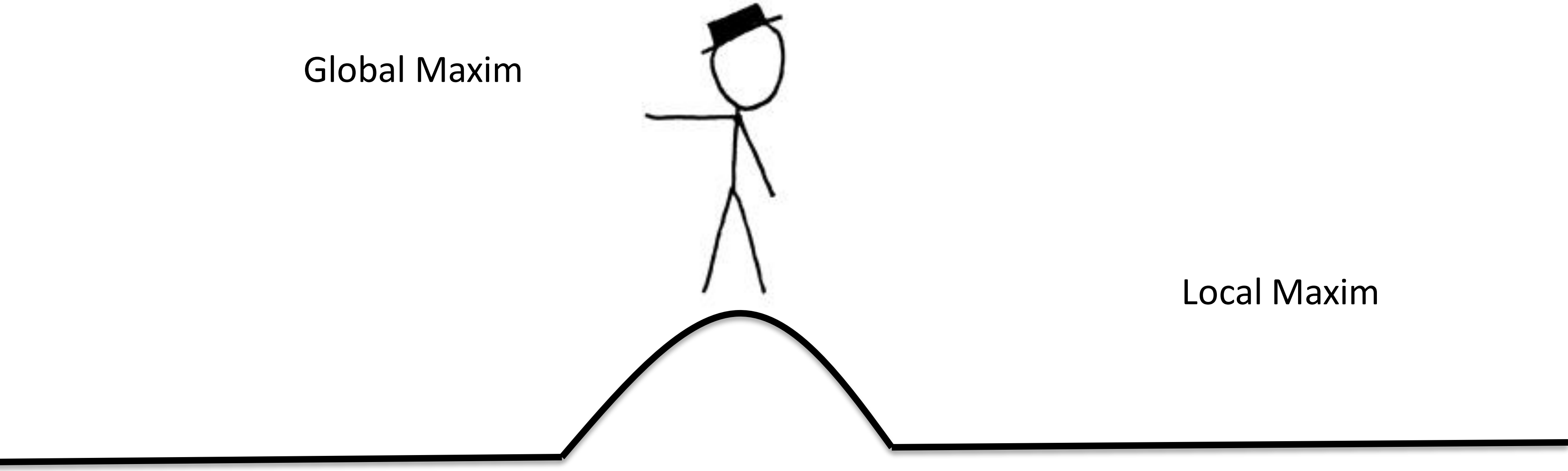
A STRATEGY FOR ARCHITECTURE EXPLORATION

The Agile Architect’s Dilemma

What is the **least** amount of upfront work required to design architecture **effectively**?

A very brief introduction to the science of design…

Design is an Optimization Problem



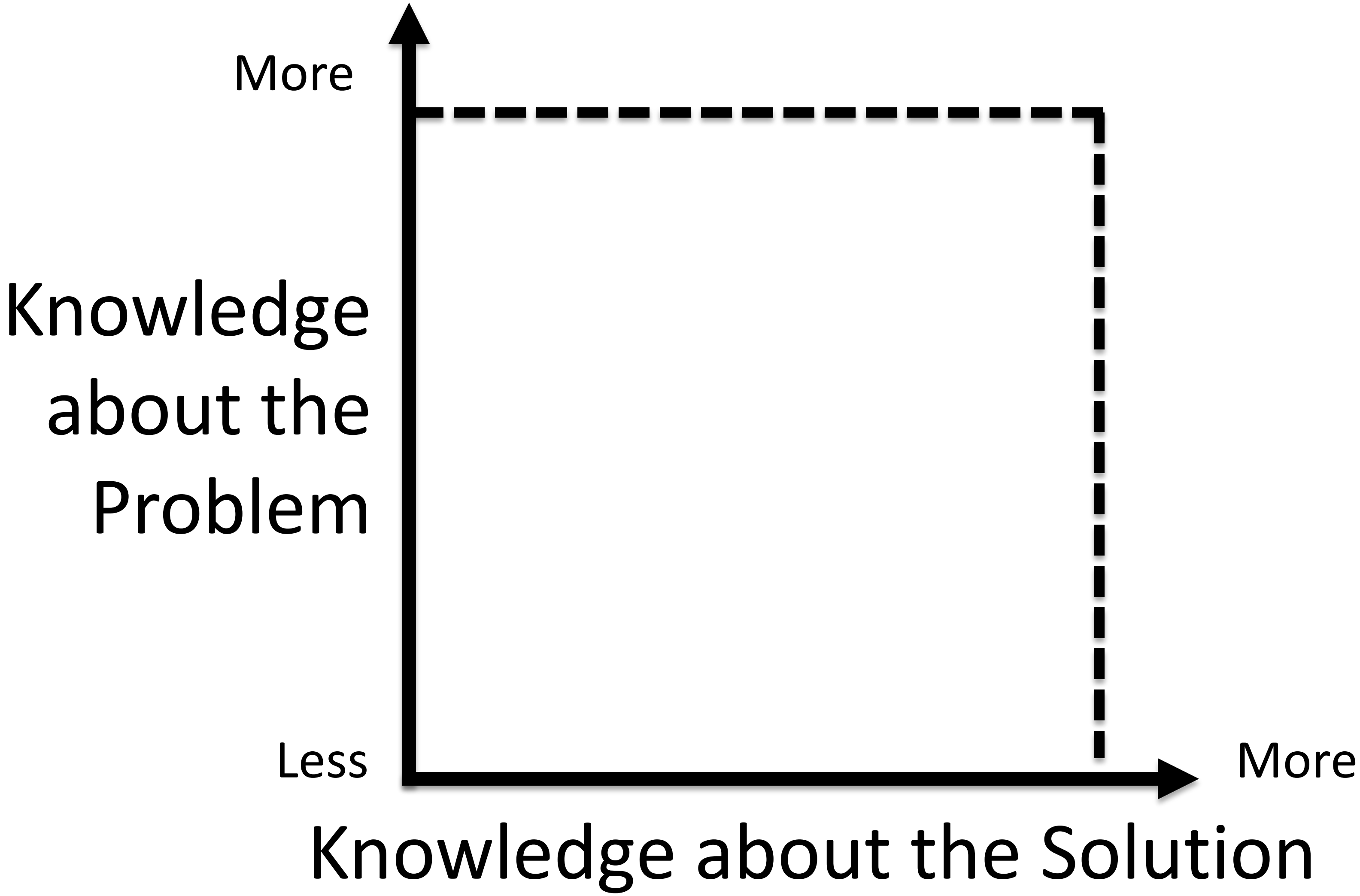
Global Maxim

Local Maxim

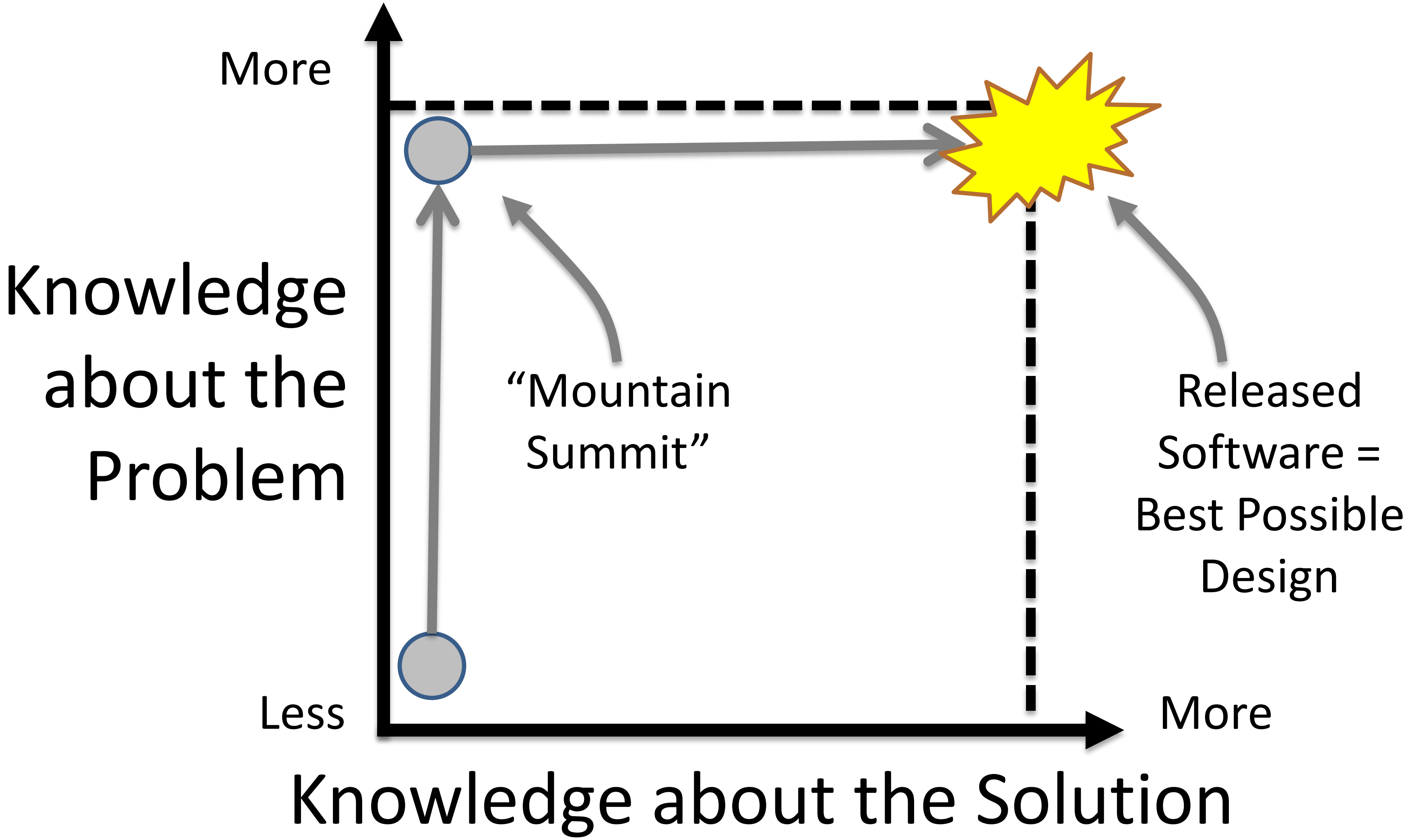
Rapid Exploration

Find the local maxims as quickly as possible… so you can see the global maxims.

The “Software Design Space”



Waterfall Exploration



More

Knowledge about the Problem

“Mountain Summit”

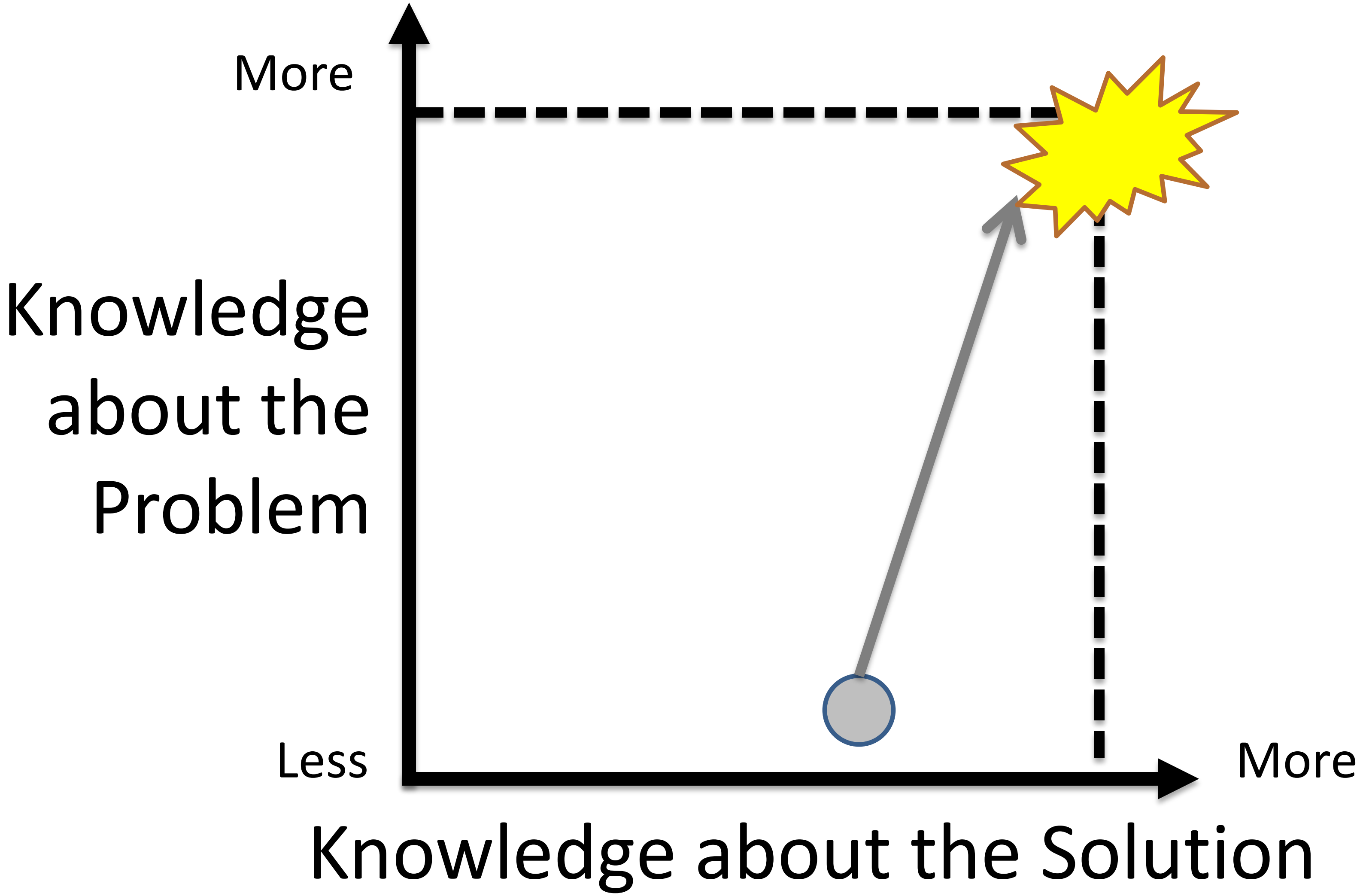
Released Software = Best Possible Design

Less

More

Knowledge about the Solution

Solution Consultant Exploration



More

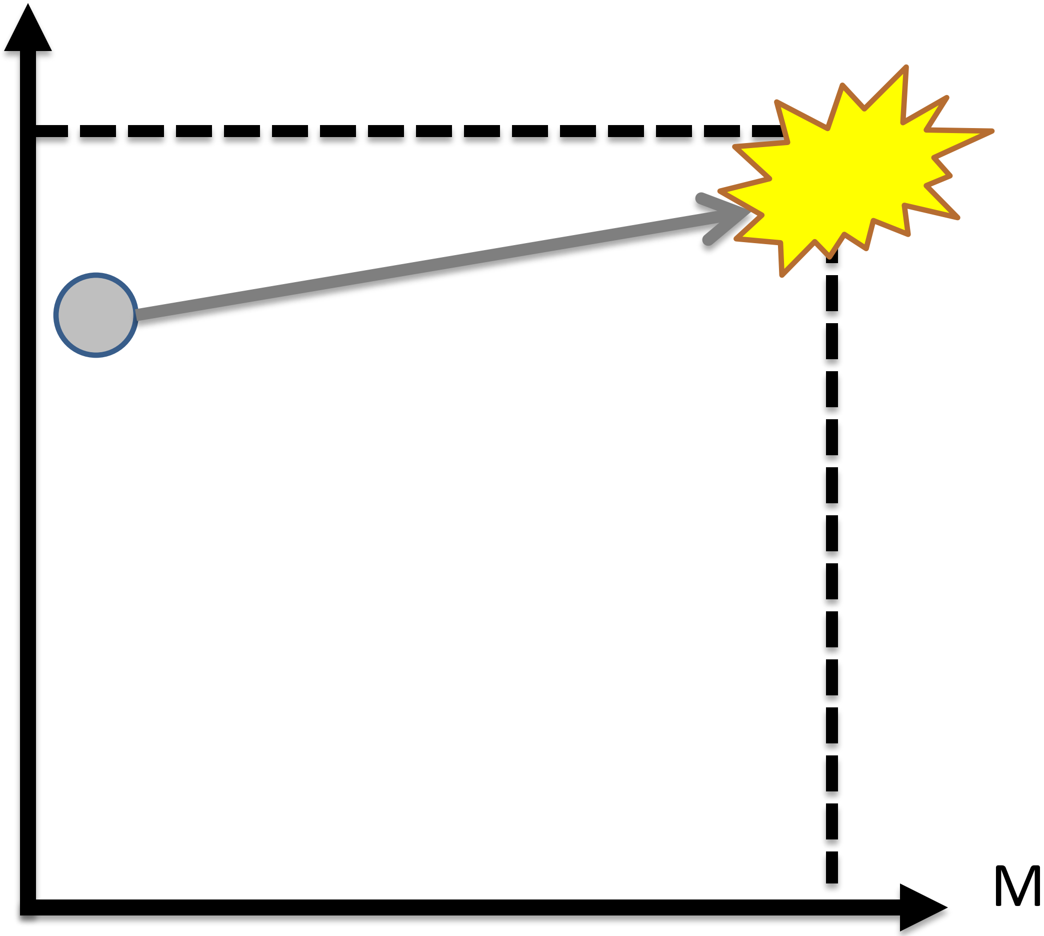
Knowledge about the Problem

Less

More

Knowledge about the Solution

Subject Matter Expert Exploration



More

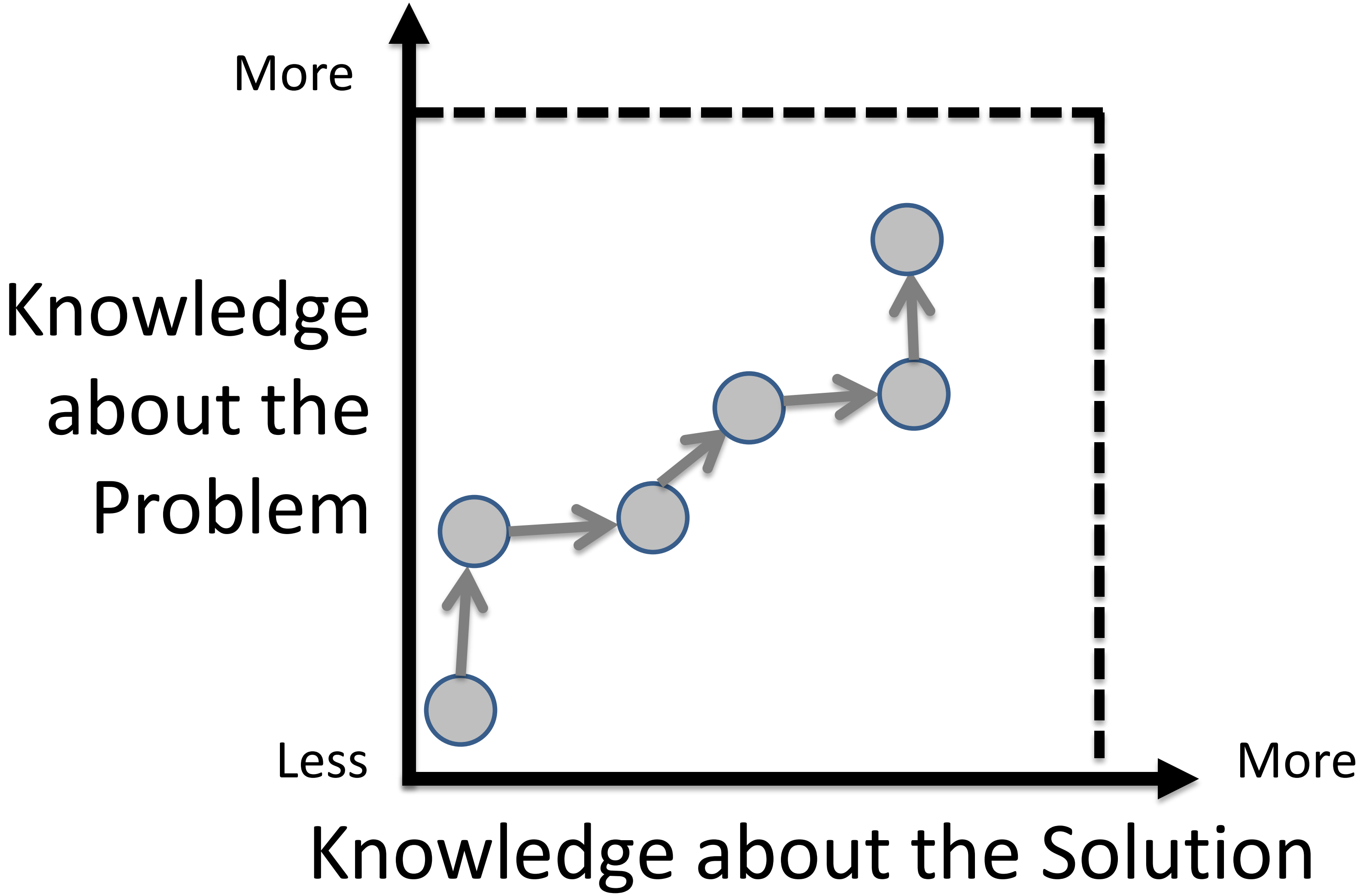
Knowledge about the Problem

Less

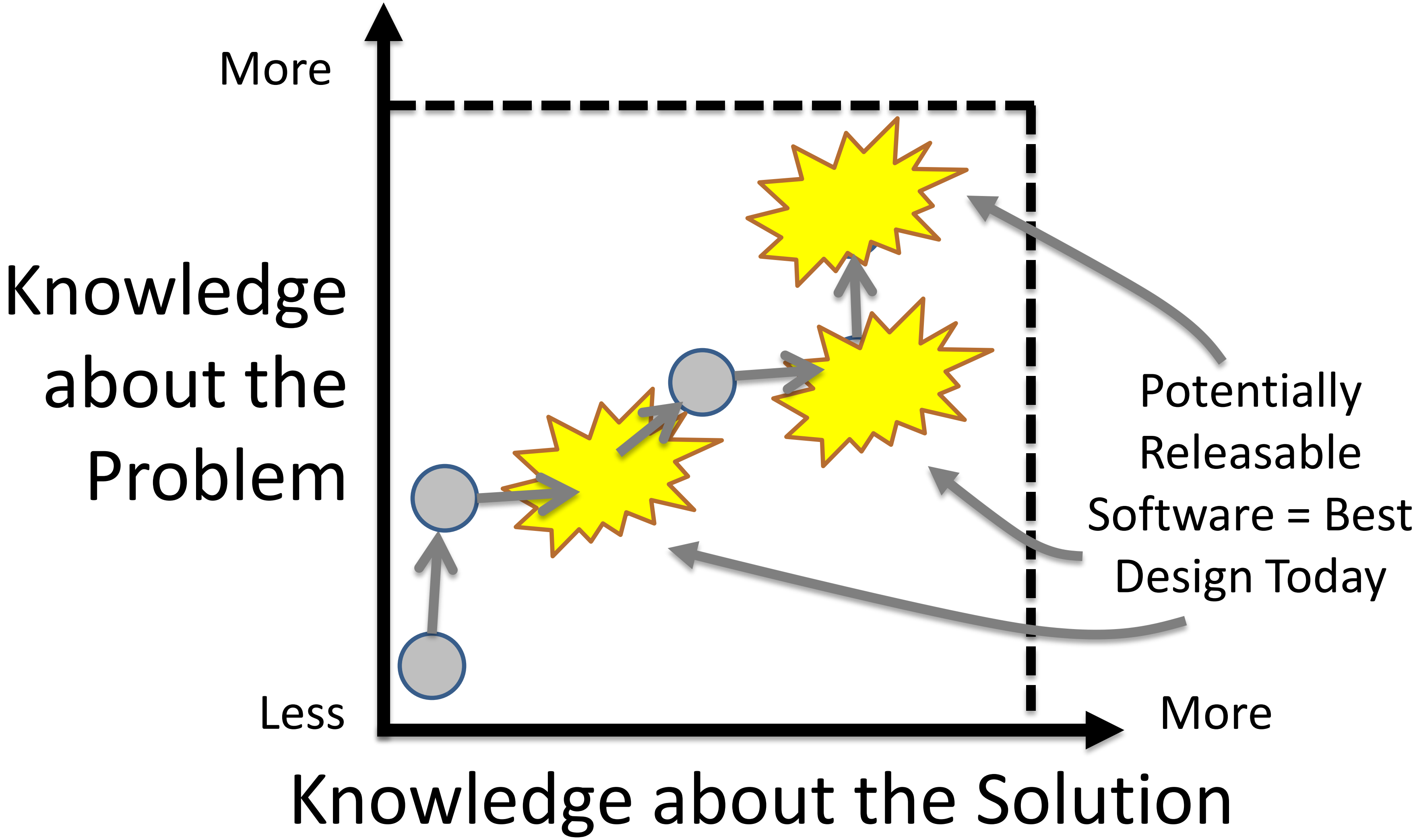
More

Knowledge about the Solution

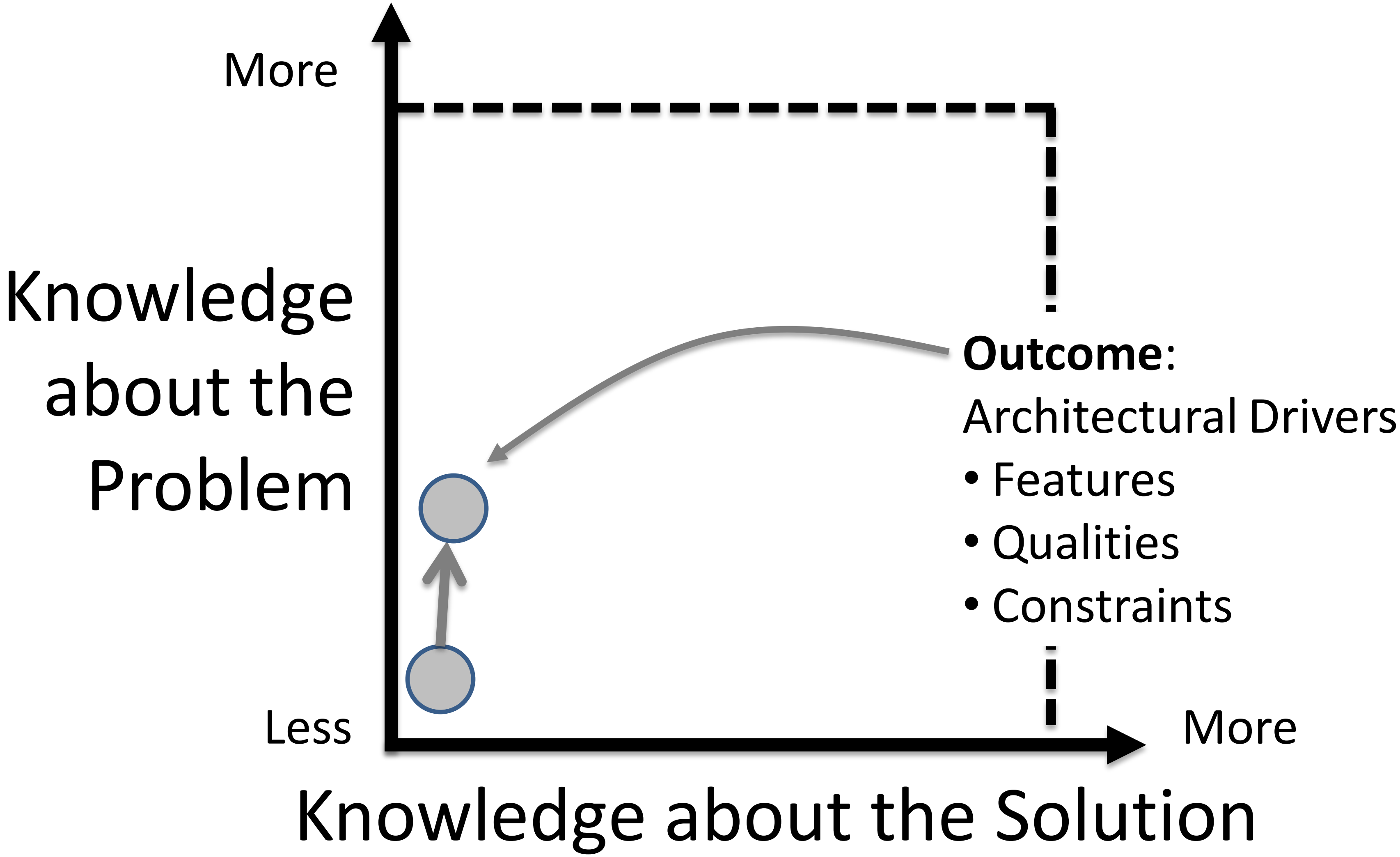
Agile Exploration



Agile Exploration


More
Knowledge about the Problem
Less
Potentially Releasable Software = Best Design Today
More
Knowledge about the Solution

Outcomes from Exploration

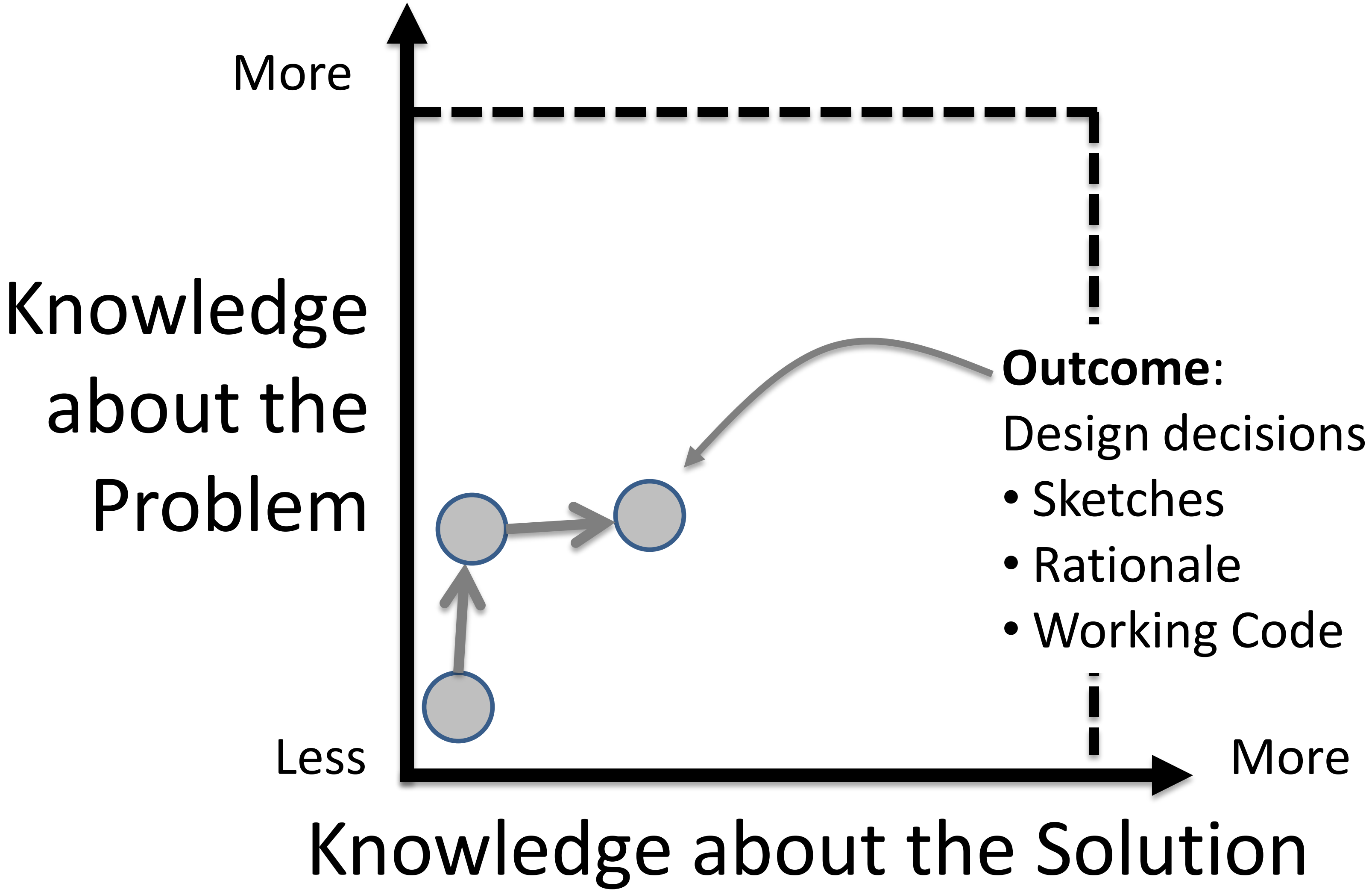


Outcome:

Architectural Drivers

- Features
- Qualities
- Constraints

Outcomes from Exploration



More

Knowledge about the Problem

Less

More

Knowledge about the Solution

Outcome:
Design decisions
- Sketches
- Rationale
- Working Code

Choose Architecture Design Strategies that …

- Embrace team values
  - – For example… reliable, adaptable, collaborative, etc.
- Explore with purpose
  - – Solution vs. Problem

RAPID EXPLORATION WORKSHOP

Four Activities to Try

- System Properties Web
- Stakeholder Map
- Round-Robin Design
- Risk Storming

A Tapas Workshop…

Challenge and Context

Your table is now a team that has been hired to build some software…

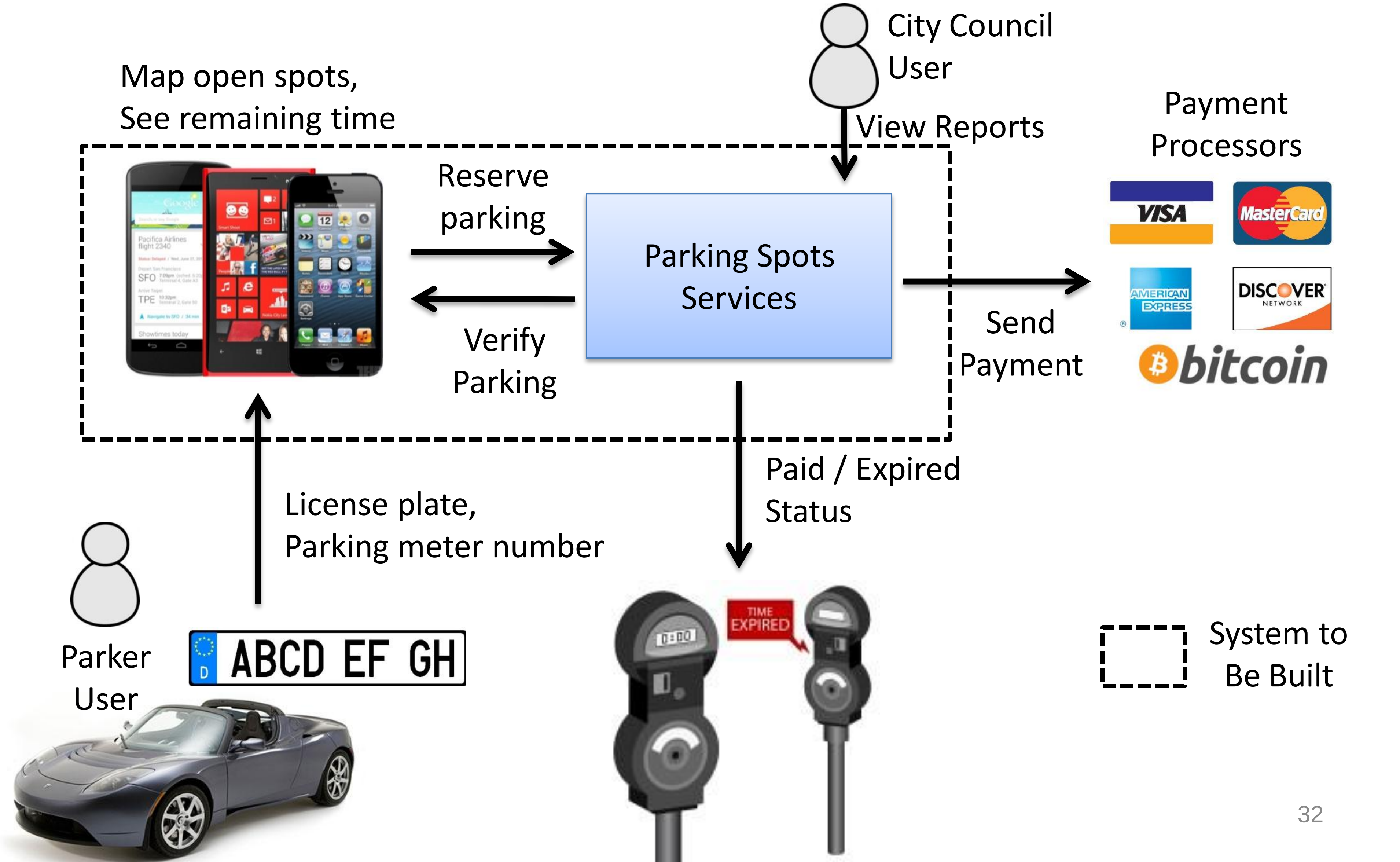
Challenge

The city of Vienna has hired you to architect a mobile application to help people find and pay for parking (for cars).

Parking App – High Level Features

- As a car driver I can…
  - Find available parking places
  - Pay to park
  - Review and pay parking tickets
- As a policeman I can…
  - Issue parking tickets
- As a city council member I can…
  - Review historical parking data and metrics

A Context Diagram…



City Council User

View Reports

Map open spots,
See remaining time

Reserve parking

Verify Parking

Parking Spots Services

Payment Processors

Send Payment

Paid / Expired Status

License plate,
Parking meter number

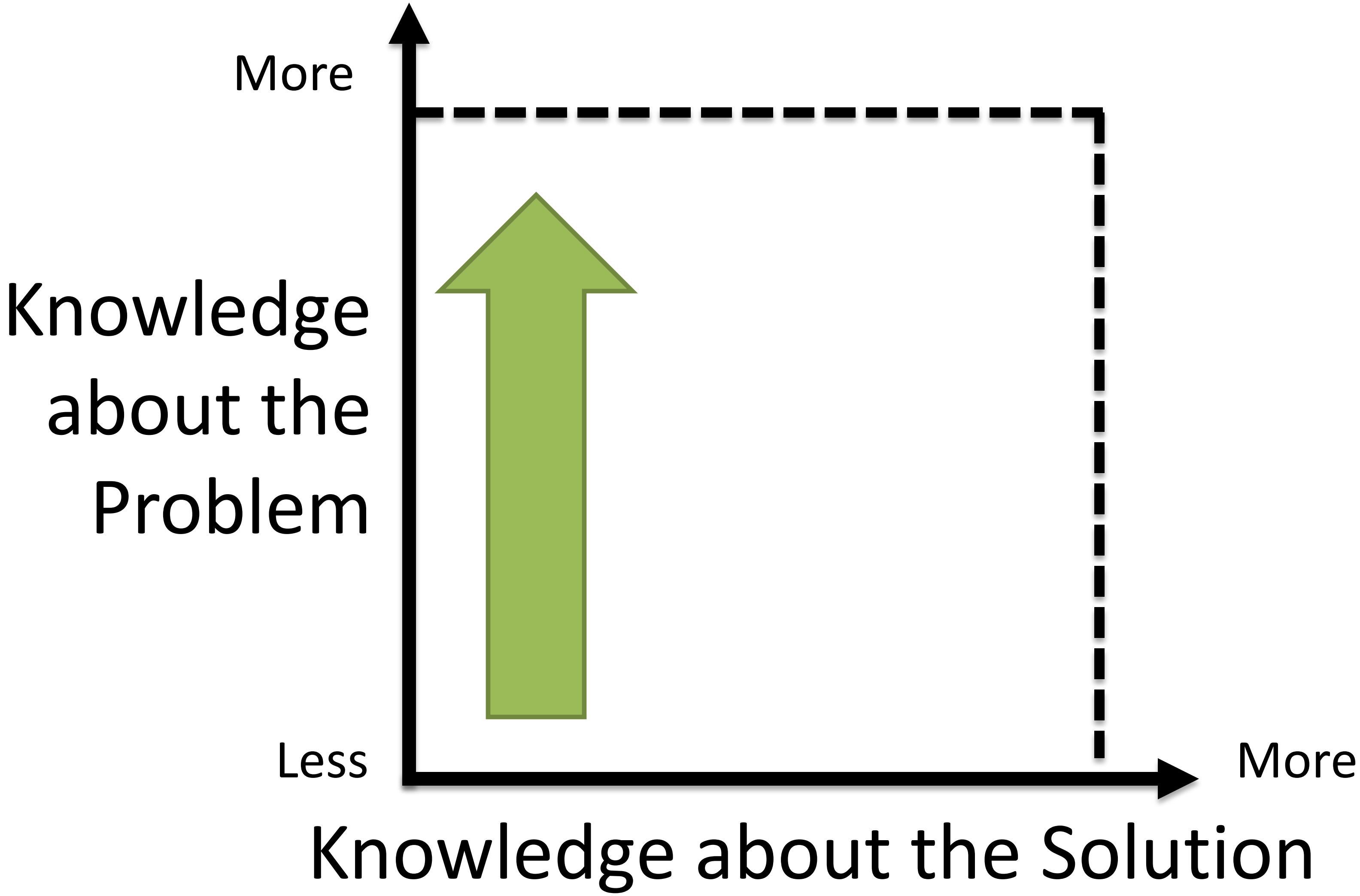
Parker User

System to Be Built

Station 1

System Properties Web

System Properties Web



More

Knowledge about the Problem

Less

More

Knowledge about the Solution

Upgrade-ability
Reusability
Availability
9
Manageability
Reliability
4
Security
6
Sprint 2
- customer
access
Crawl-ability
1
24/7
Upgrade to
CXD
Query-ability
1
Build-ability
Deploy-ability
1
Maintainability
Scalability
Modifiability
2

System Properties Web

Description Help stakeholders to collaboratively generate, affinity cluster, and prioritize raw quality attribute scenarios

Time Needed 1 – 3 hours (depending on the web size)

Benefits

- Focus on system qualities over functions/feature
- Visually show how two systems differ by looking at quality attributes

Participants As many relevant stakeholders as available – team, customer, IT, etc.

System Properties Web Station Instructions

Objective: Identify raw quality attributes ("-ilities") for the Vienna Parking App.

Steps:

1. Review quality requirements on web.
2. Brainstorm stakeholder concerns (~3 minutes).
3. Read concerns aloud and categorize on web.
4. Observe and reflect as a group

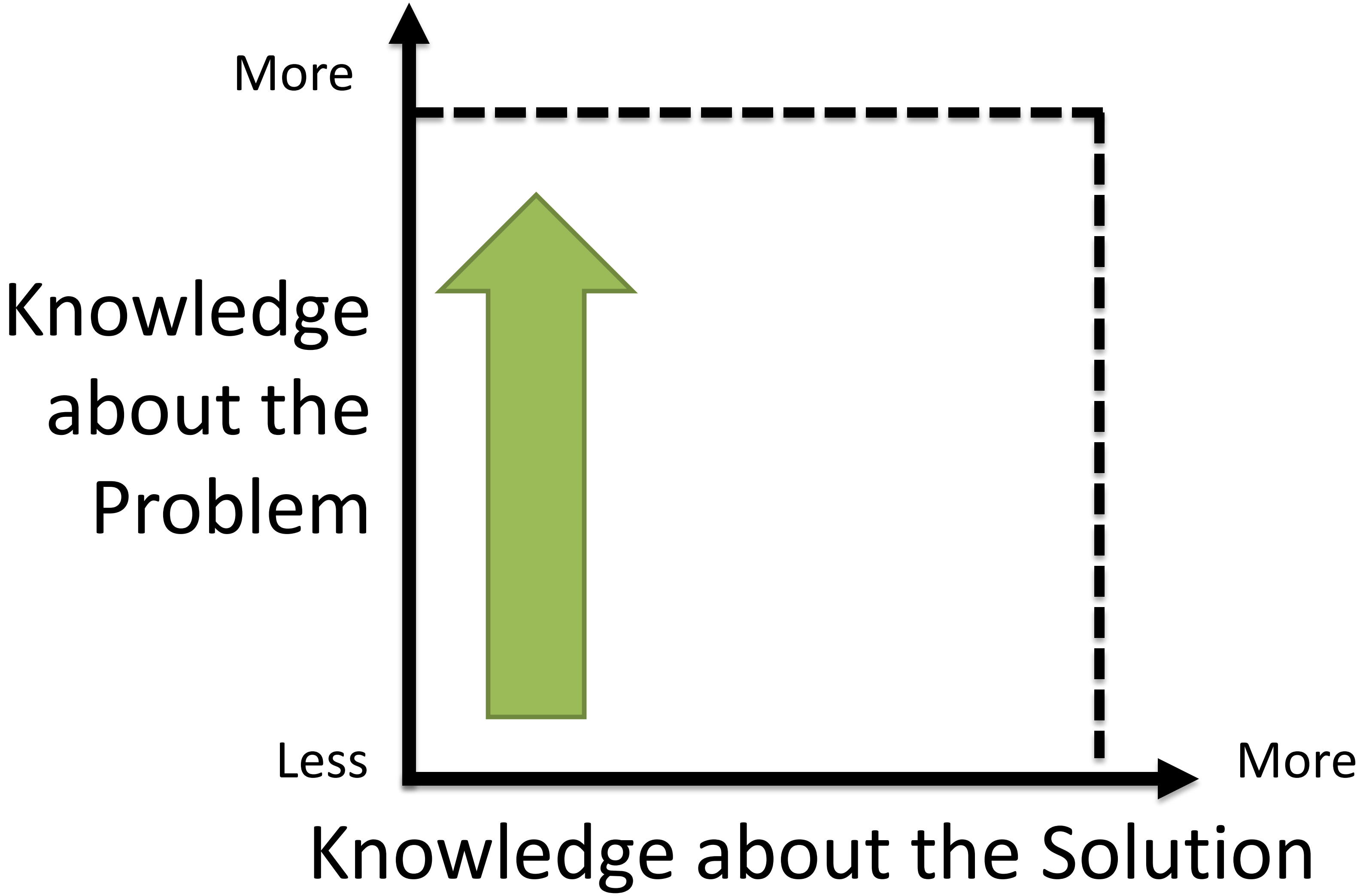
Guidelines and hints:

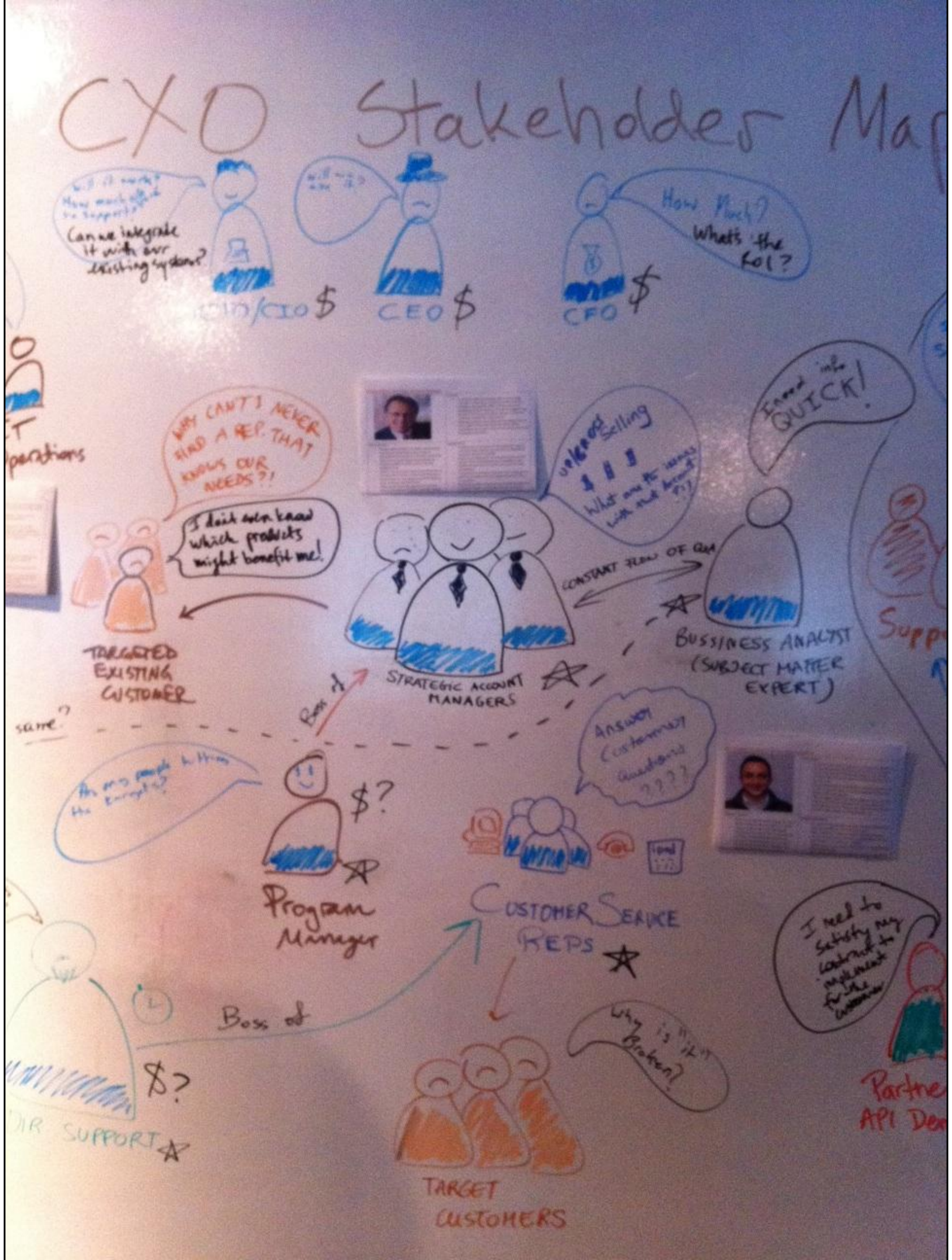
- Write 1 scenario/concern per sticky note
- Everyone writes at least 1 sticky note

Station 2

Stakeholder Map

Stakeholder Map




CXO Stakeholder Map
Can we integrate it with our existing systems?
[illegible]/CIO $
CEO $
How Much? What's the ROI?
CFO $
T
Operations
WHY CAN'T I NEVER FIND A REP. THAT KNOWS OUR NEEDS?!
I dont even know which products might benefit me!
TARGETED EXISTING CUSTOMER
Selling
What are the [illegible] with [illegible] ?!?
I need info QUICK!
CONSTANT FLOW OF Q&A
STRATEGIC ACCOUNT MANAGERS
BUSSINESS ANALYST (SUBJECT MATTER EXPERT)
Supp
same?
Boss of
$?
Program Manager
Answer Customer Questions ????
CUSTOMER SERVICE REPS
I need to satisfy my contract to implement for the customer
Boss of
$?
IR SUPPORT
Why is "it" Broken?
TARGET CUSTOMERS
Partne
API Dev

Stakeholder Map

Description A network diagram of the people involved with or impacted by a given system or system design

Time Needed 30 – 45 minutes

Benefits

- Identify more than the usual stakeholders
- Document, guide plans for research
- Keep the team focused on people rather than technologies

Participants As many potential stakeholders as available – team, customer, etc.

Stakeholder Map Station Instructions

Objective: Visualize the relationships, hierarchies, and interactions between all the people who have an interest in the system to be built.

Steps:

1. Add and annotate stakeholders collaboratively until time runs out or the map seems complete (for now).

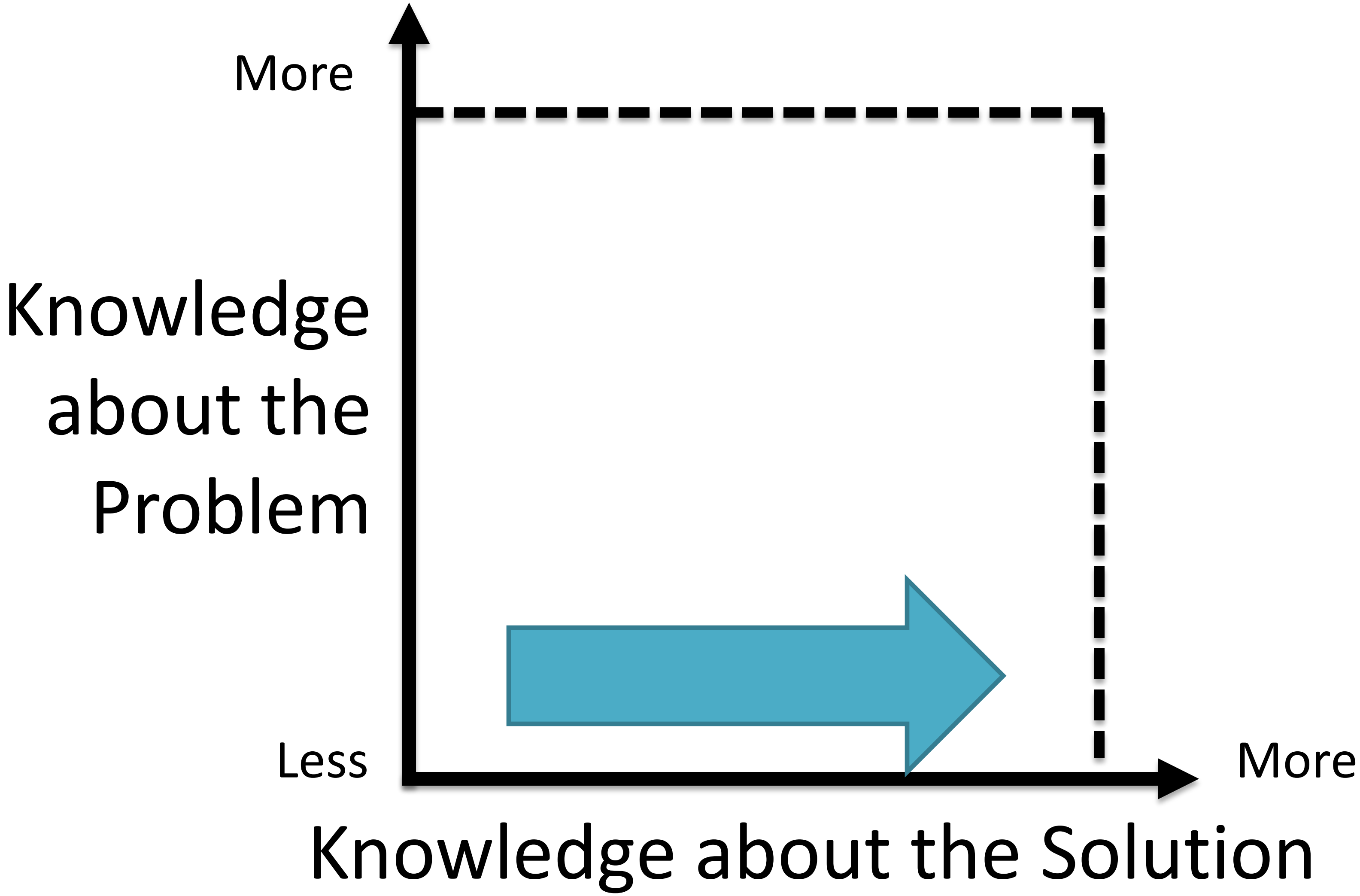
Guidelines and hints:

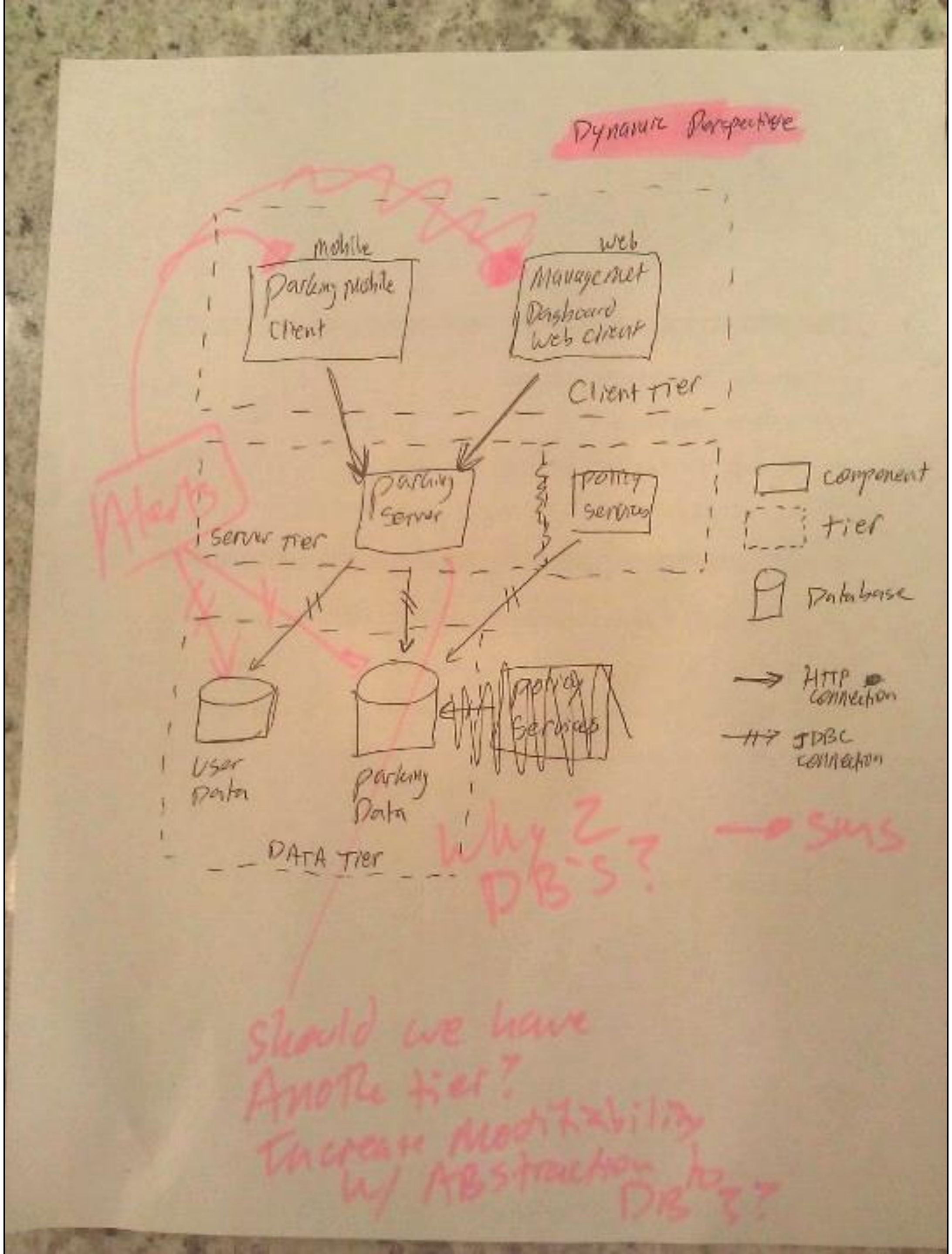
- Simple icons to represent individual people
- Label people by **specific** role
- Don’t represent categories of people as a single icon
- Speech bubbles represent thoughts, feelings
- Arrows connect people
- Label lines to describe relationships

Station 3

Round Robin Design

Round-Robin Design




Dynamic Perspective
mobile
web
Parking mobile Client
Management Dashboard Web client
Client Tier
Parking Server
policy service
Server Tier
Alerts
User Data
Parking Data
~~policy Service~~
DATA Tier
component
tier
Database
HTTP connection
JDBC connection
SMS
Why 2 DB's?
Should we have Another tier?
Increase Modifiability w/ ABstraction to DB's?

Round Robin Design

Description Quickly generate and vet many architecture design ideas through quick succession of fast peer reviews

Time Needed 60 – 90 minutes

Benefits

- Foster creativity by constraining design
- Create opportunities for unplanned combinations.
- Encourage group ownership of the design
- Build consensus among disparate ideas.

Participants Team
(other stakeholders can help validate)

Round Robin Design Station Instructions

Objective: Quickly build a collection of ideas architecting the system, then converge ideas and start to build consensus.

Steps:

1. Design the **physical** (i.e. allocation) architecture.
2. Everyone sketches a design (~5 minutes)
3. Give your design to the person on the left.
4. Critique and annotate the design (~3 minutes)
5. Return papers to original author
6. Briefly discuss insights

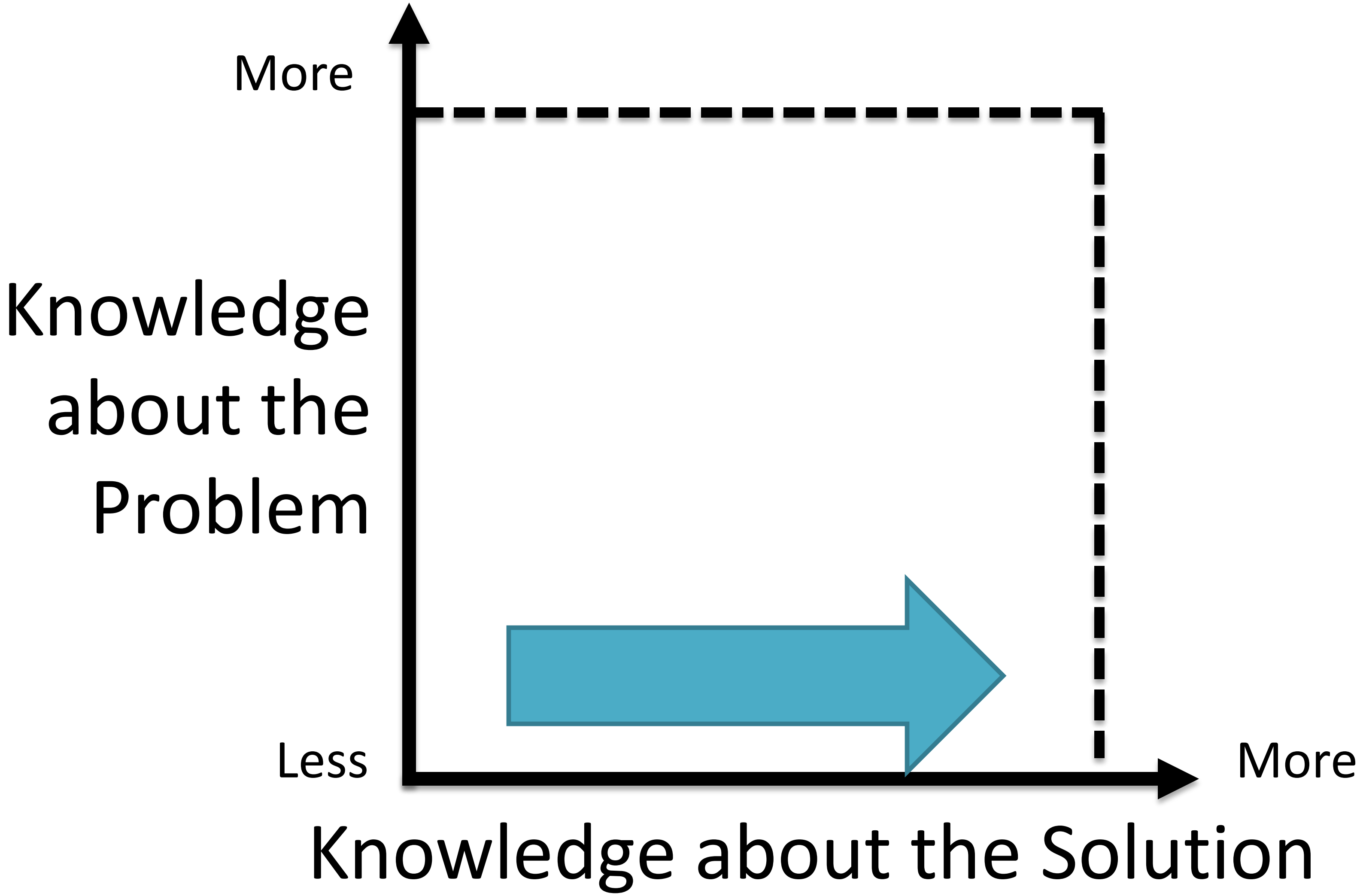
Guidelines and hints:

- Everyone sketches
- No right or wrong answers
- Use different color ink for each critique

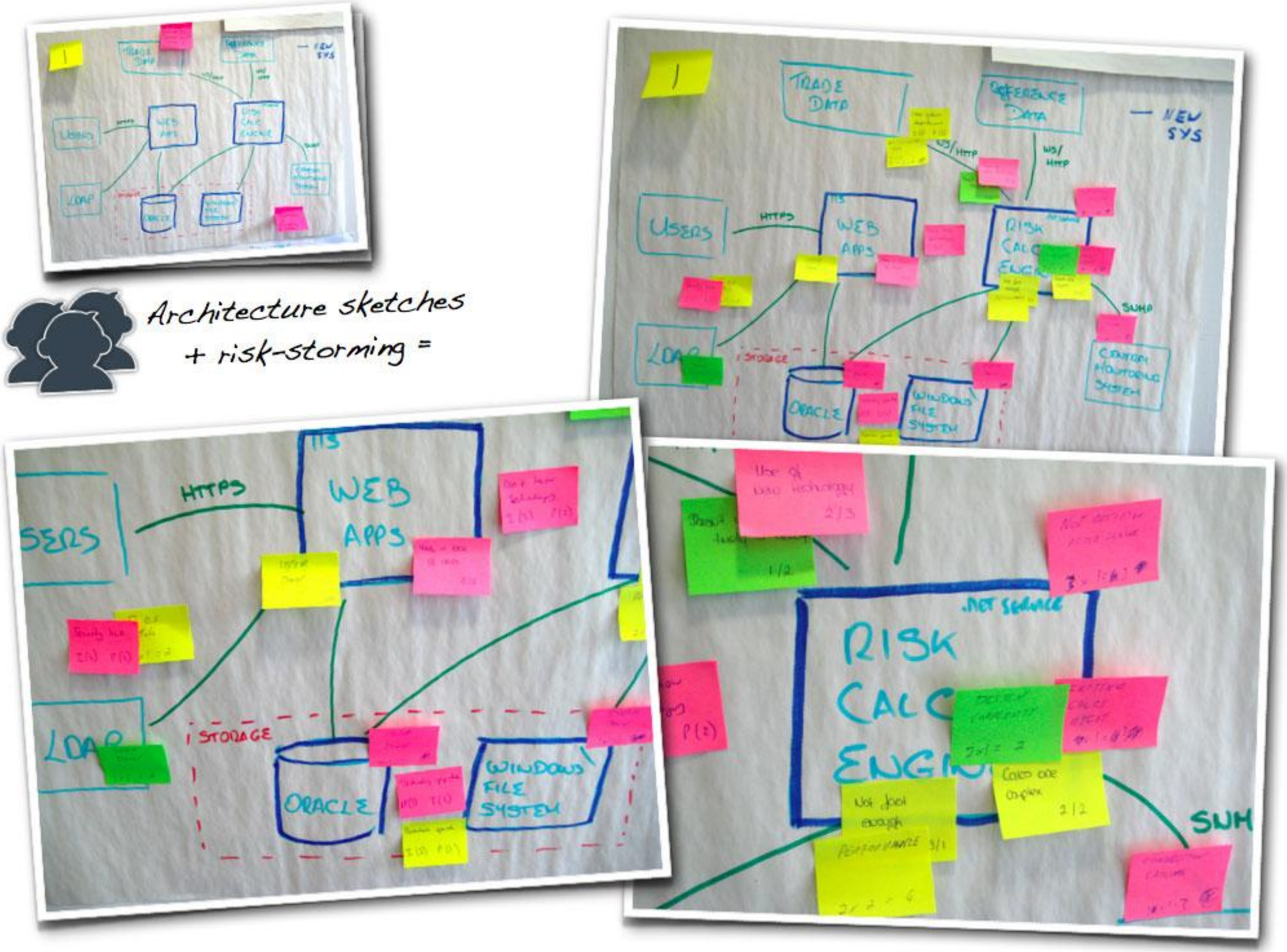
Station 4

Risk Storming

Risk Storming



Risk Storming



http://www.codingthearchitecture.com/2012/07/11/risk_storming.html

Risk Storming

Description Use the architecture to kick starting brainstorming risks and visualize “trouble spots” in architecture views

Time Needed 30 – 90 minutes

Benefits

- Identify risks in the proposed architecture.
- Visualize “troubling” parts of the system
- Constrain risk identification to only architectural concerns

Participants Team, relevant stakeholders

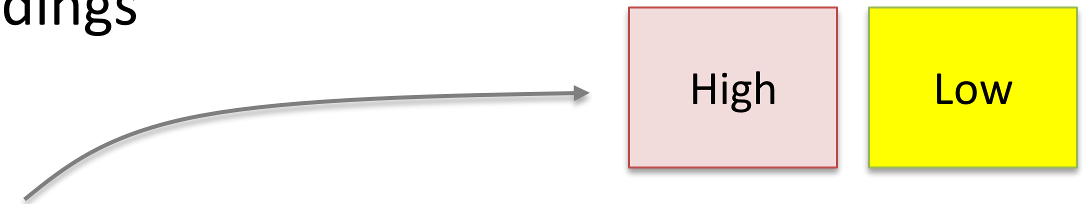
See: http://www.codingthearchitecture.com/2012/07/11/risk_storming.html
by Simon Brown

Risk Storming
Station Instructions

Objective: Identify and prioritize architectural risks so that a suitable mitigation strategy can be identified.

Steps:

1. Review provided architecture view
2. Brainstorm risks, write one per sticky note
3. Place sticky notes on the most relevant area of the sketches
4. Reflect and review findings



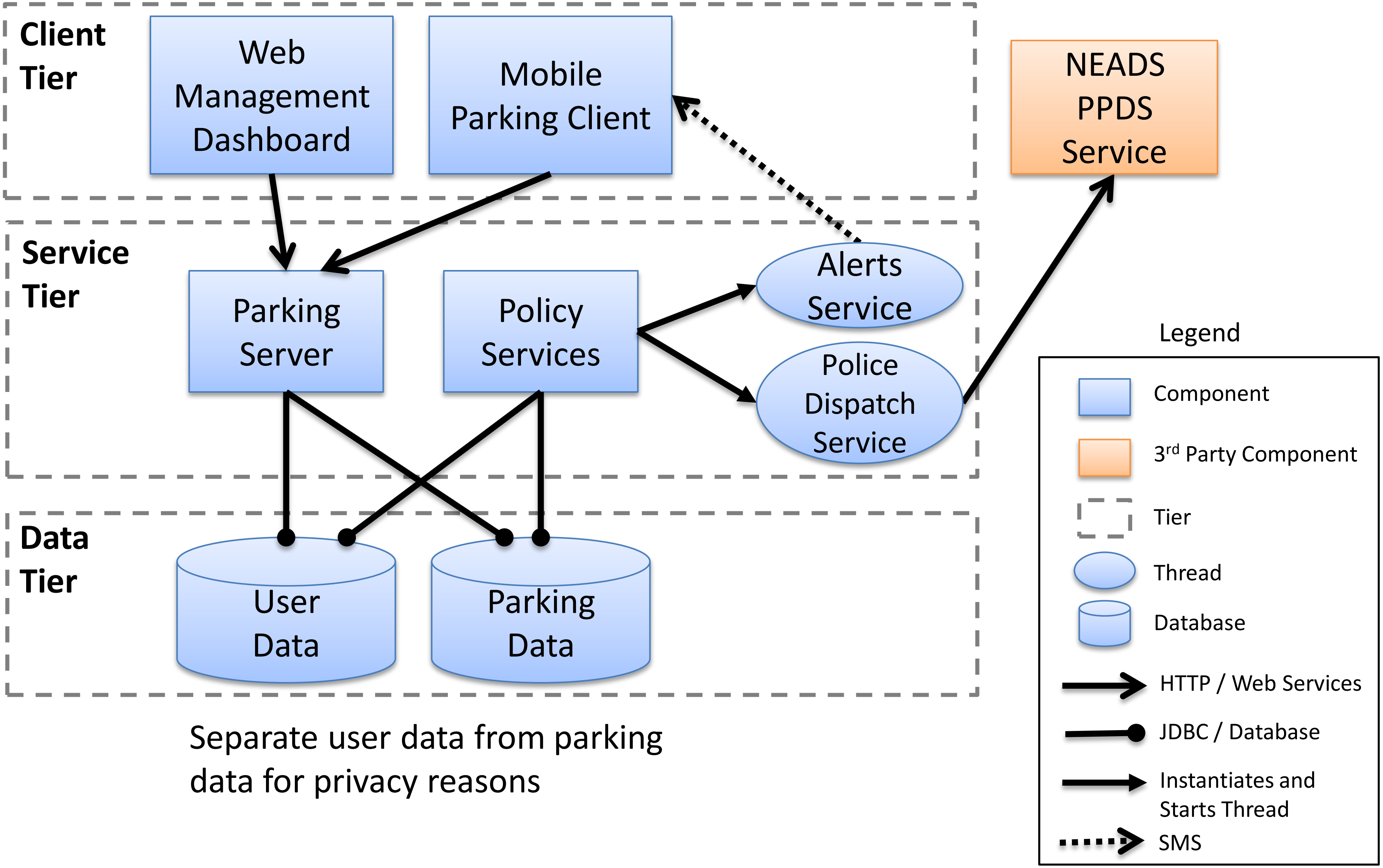
High

Low

Guidelines and hints:

- Different colored sticky notes indicate high/low "exposure"
- Look for clusters of risk or "trouble spots"
- Mitigate by designing experiments or doing research
- Variants: Rose-Bud-Thorn, Question-Fact-Idea

Mobile Parking System (Dynamic Perspective)



Separate user data from parking data for privacy reasons

WORKSHOP!

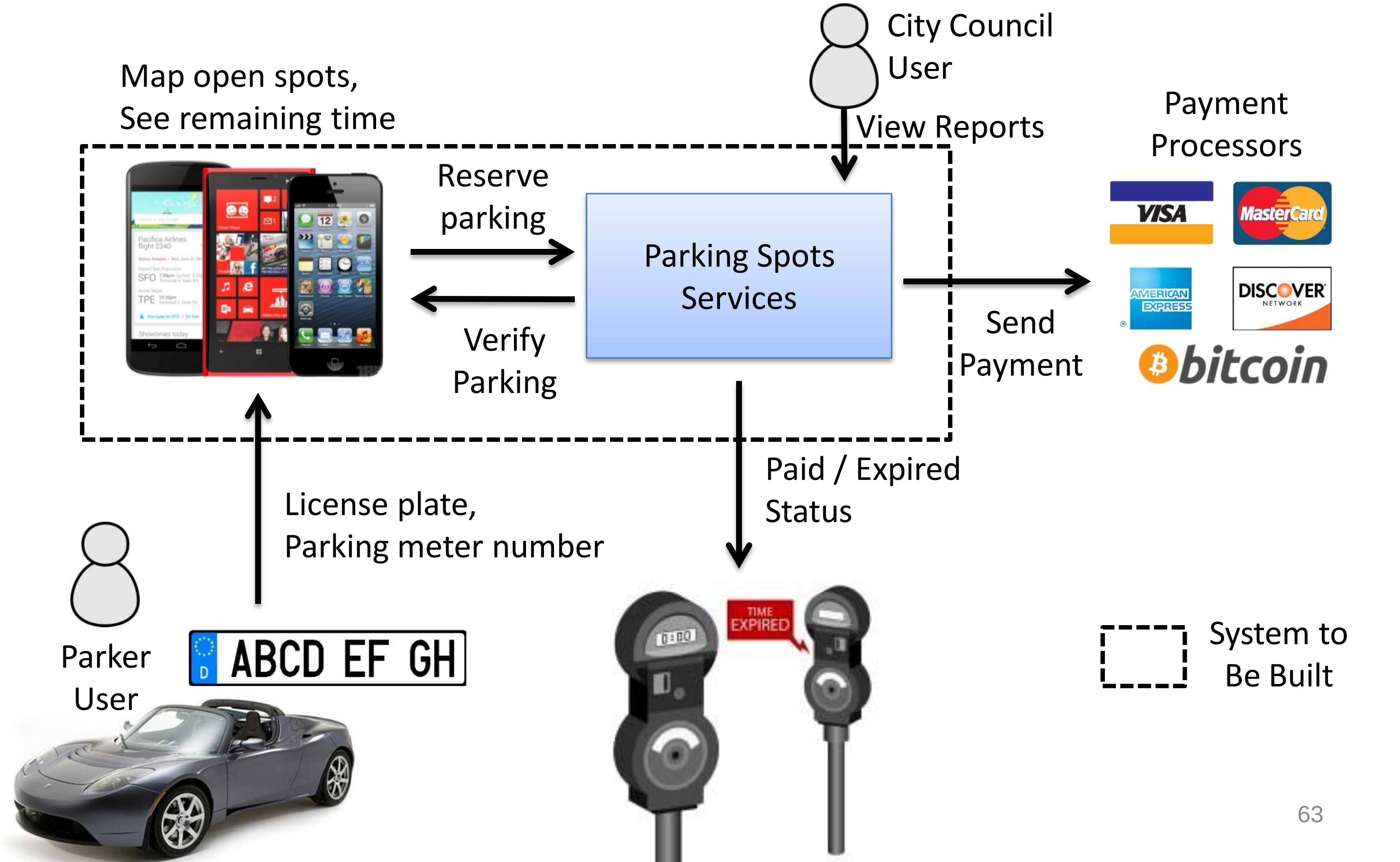
Ground Rules

- Four activities – try to do at least 3
  - Work in small groups (people at your table)
  - Any order
- No right or wrong answers
  - Use your imagination to fill in missing details
- Watch the clock (I'll help too)
  - Switch every ~10 minutes
- Ask me questions if you need help or clarification
- HAVE FUN! ☺
- Tweet / Share your experiences!
  - #xp2013
  - #rapidarchitecture

Challenge

The city of Vienna has hired you to architect a mobile application to help people find and pay for parking (for cars).

A Context Diagram…



WRAP-UP

Silver Toolbox

Some Questions for Reflection

- Did certain practices need to be applied in a specific order?
  - What if this wasn't a simulation?
- Which activity was your favorite?
- Was there an activity you hated?
- Would you use any of these activities if the system was…
  - Really big?
  - Really risky?
  - The same old thing we built last year?

Sharing Your Architecture

- System Metaphor
- Architecture Haiku
- Cartoons
- Low Fidelity Sketches
- Full Architecture Description

Other Workshops and Activities…

- Mini-Quality Attributes Workshop
- Mini-Software Risk Evaluation Workshop
- Design the Extremes
- Tactical Planning Circle
- System Personas
- Concern, Question, Comment
- Interface Empathy Map
- More… ?

Thank you!

Michael Keeling

@michaelkeeling

http://neverletdown.net

References

- Ariadna Font, *Bringing UX and Agile Together*. http://ariadna.font.cat/
- Simon Brown, *Coding the Architecture.* http://www.codingthearchitecture.com
- Clements et al. *Documenting Software Architectures: Views and Beyond (2nd Edition)* 2010