Security in Agile Development

So, you're telling me it can be agile and secure?

Who am I?

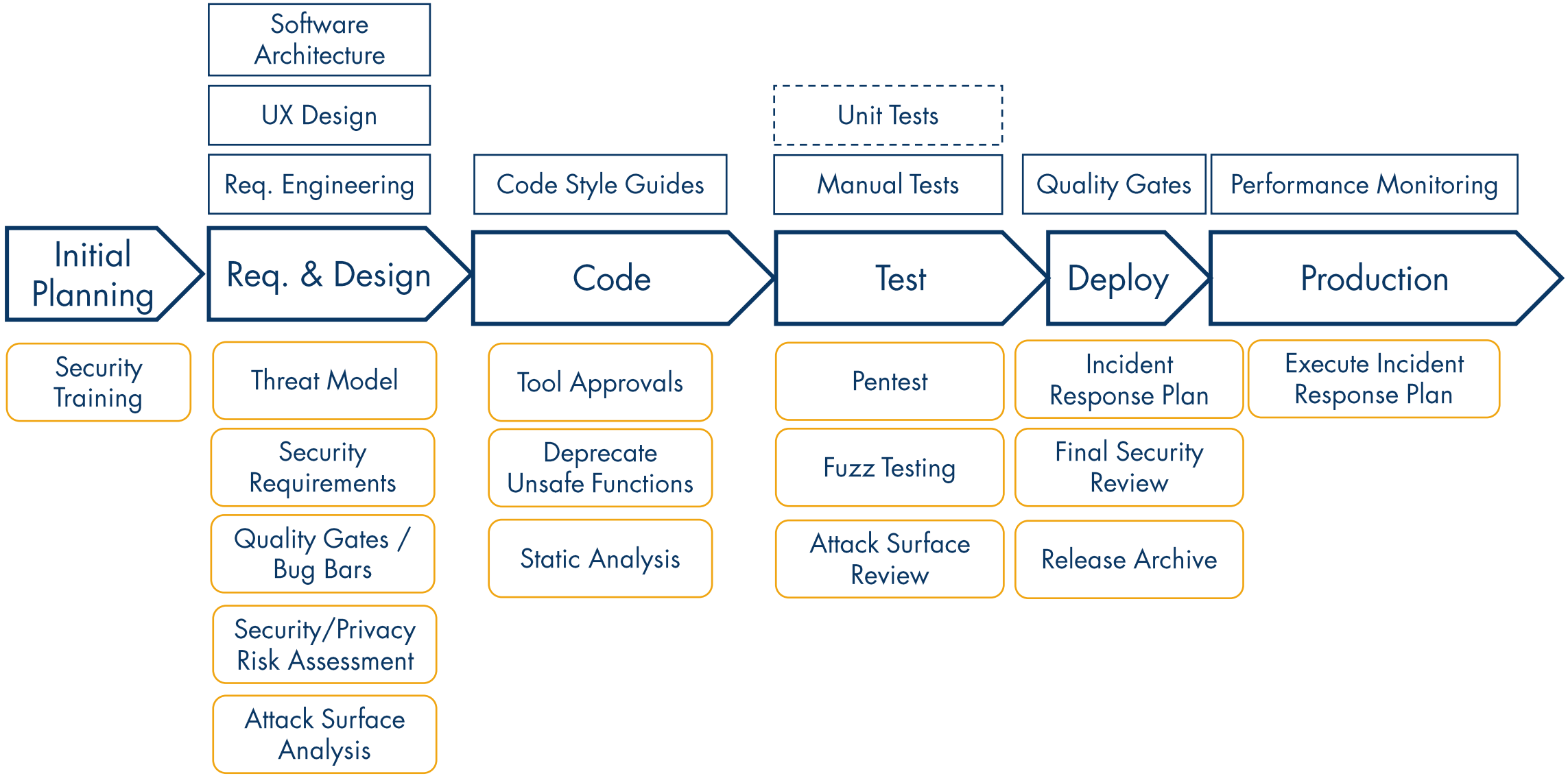
Daniel Schwarz

Senior Security Architect @ condignum GmbH in Vienna

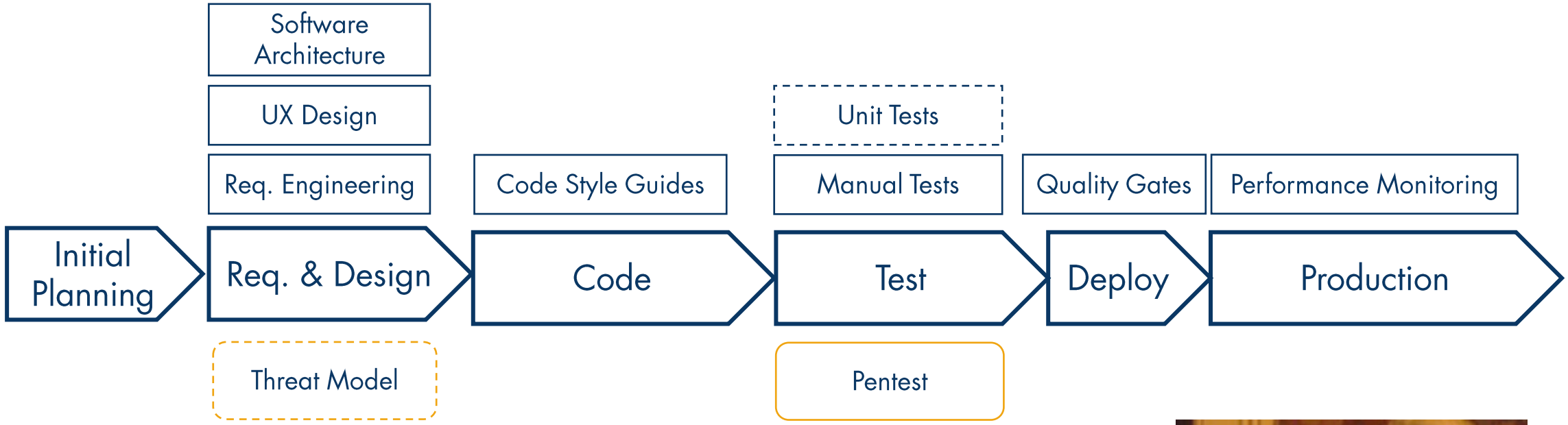
Lecturer @ St. Pölten University of Applied Sciences

- appropriate steps to ensure information security -

Good ol Waterfall times



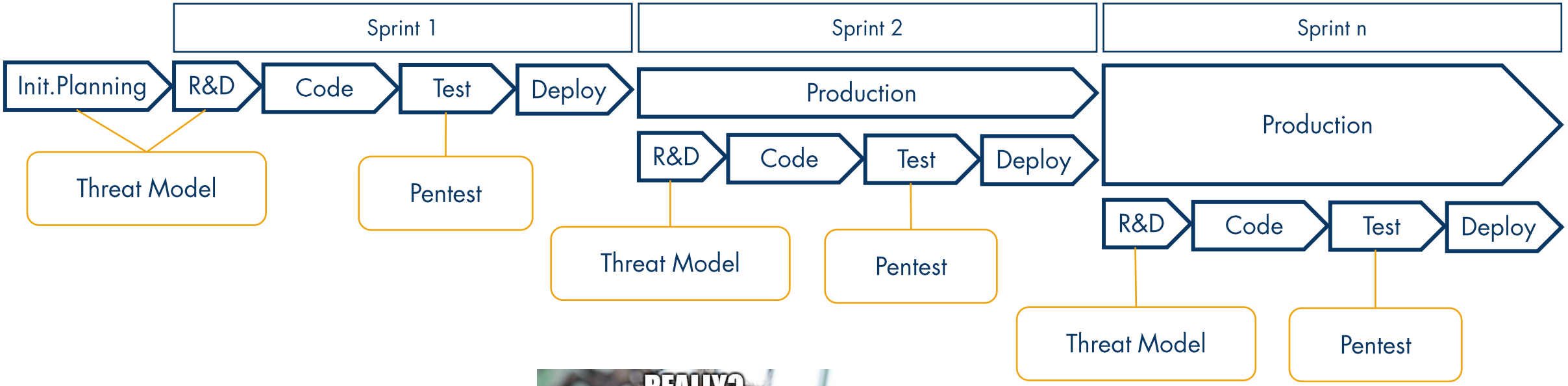
Good ol Waterfall times



Let‘s get agile

https://bitbytesoft.com/phases-of-agile-software-development-life-cycle/

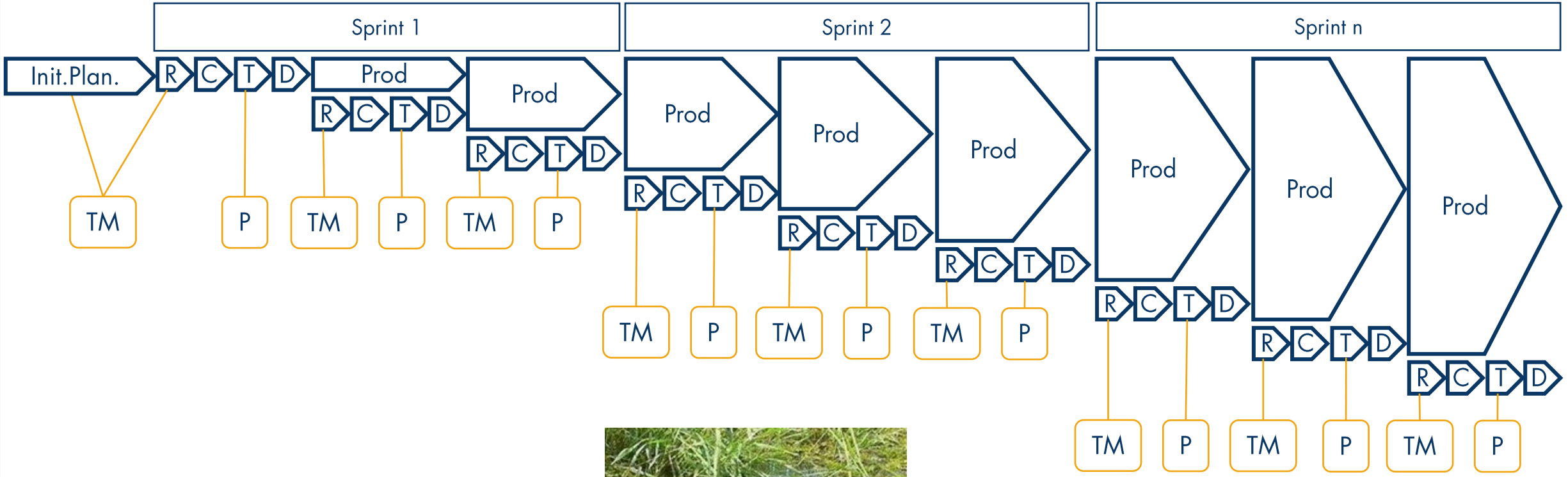
Let's get agile



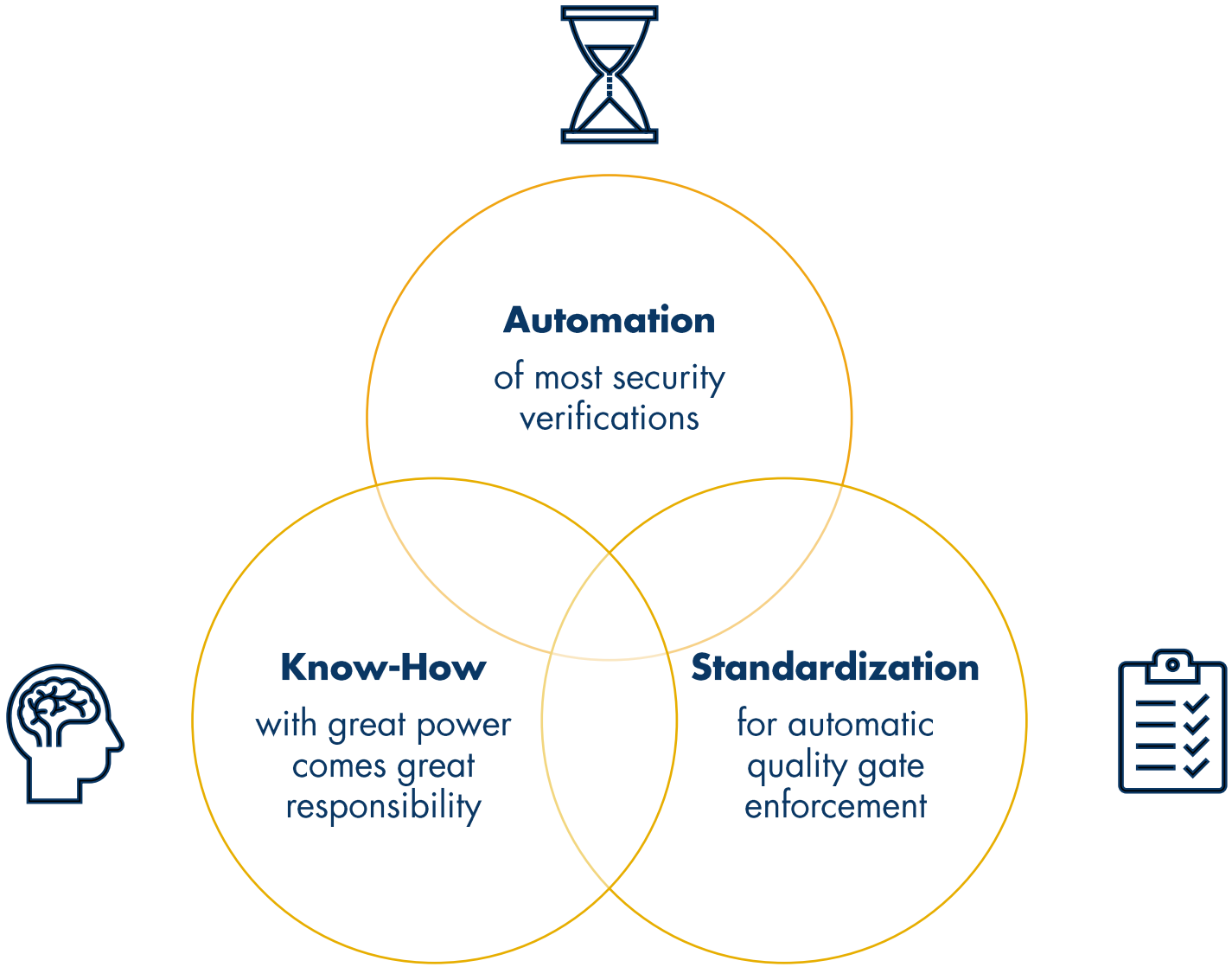
Pentest in every sprint?

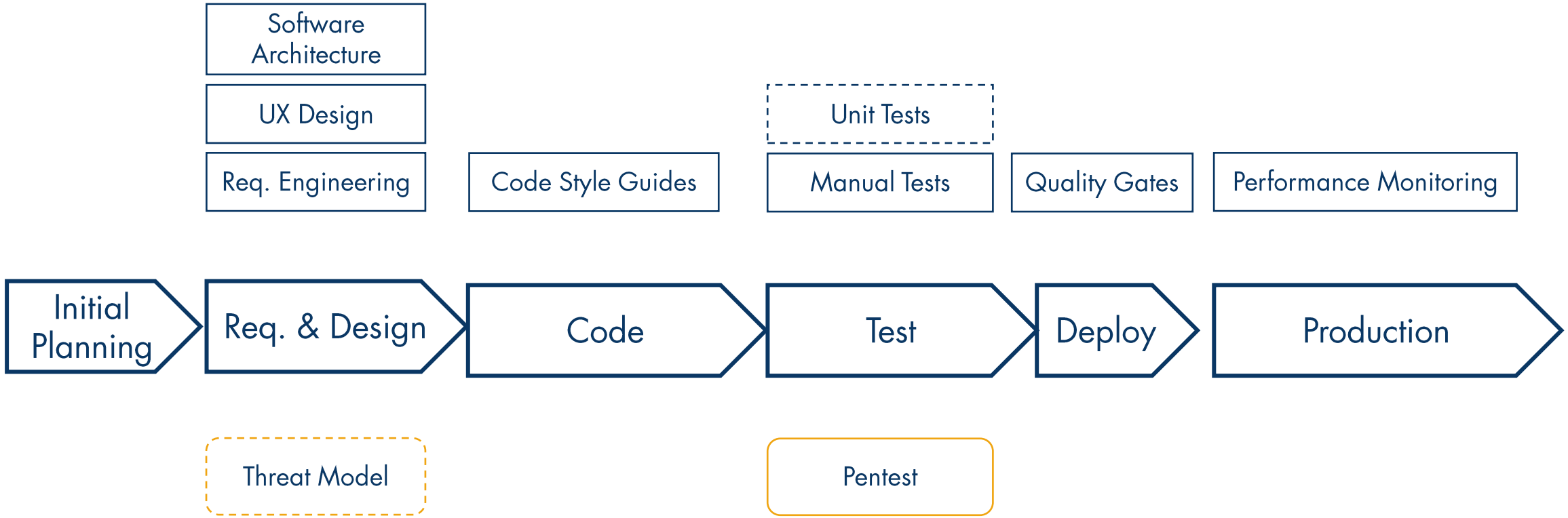
- Within a year of Amazon's move to AWS, engineers were deploying code every 11.7 seconds, on average. -

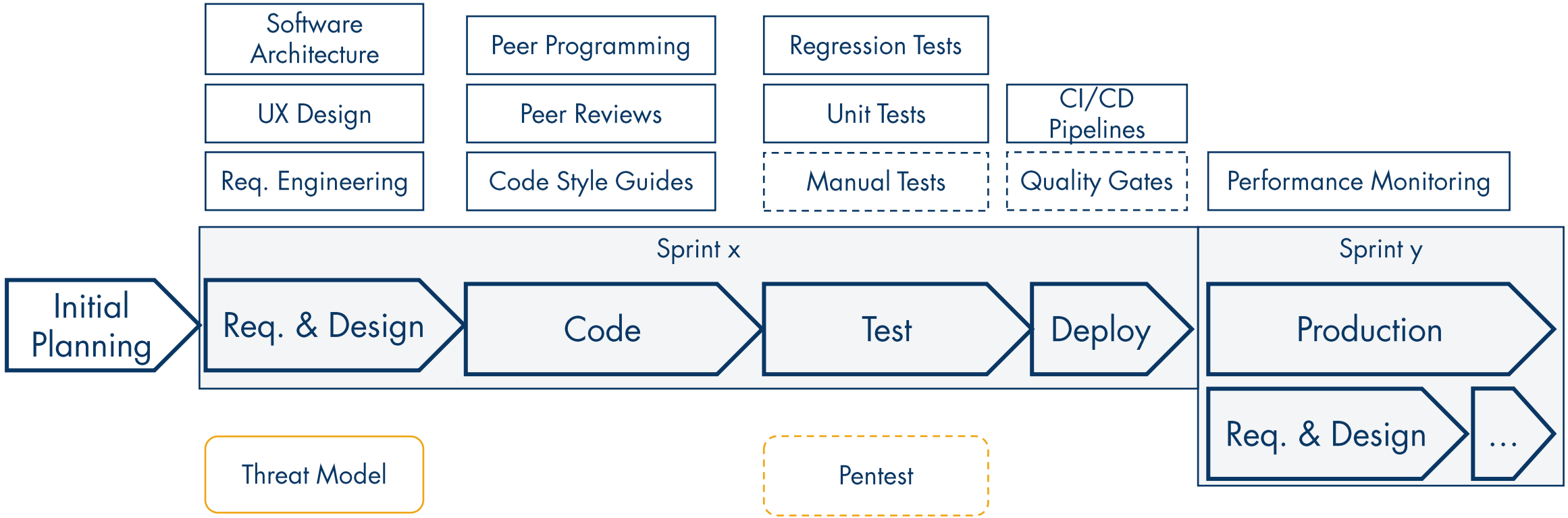
https://techbeacon.com/app-dev-testing/10-companies-killing-it-devops

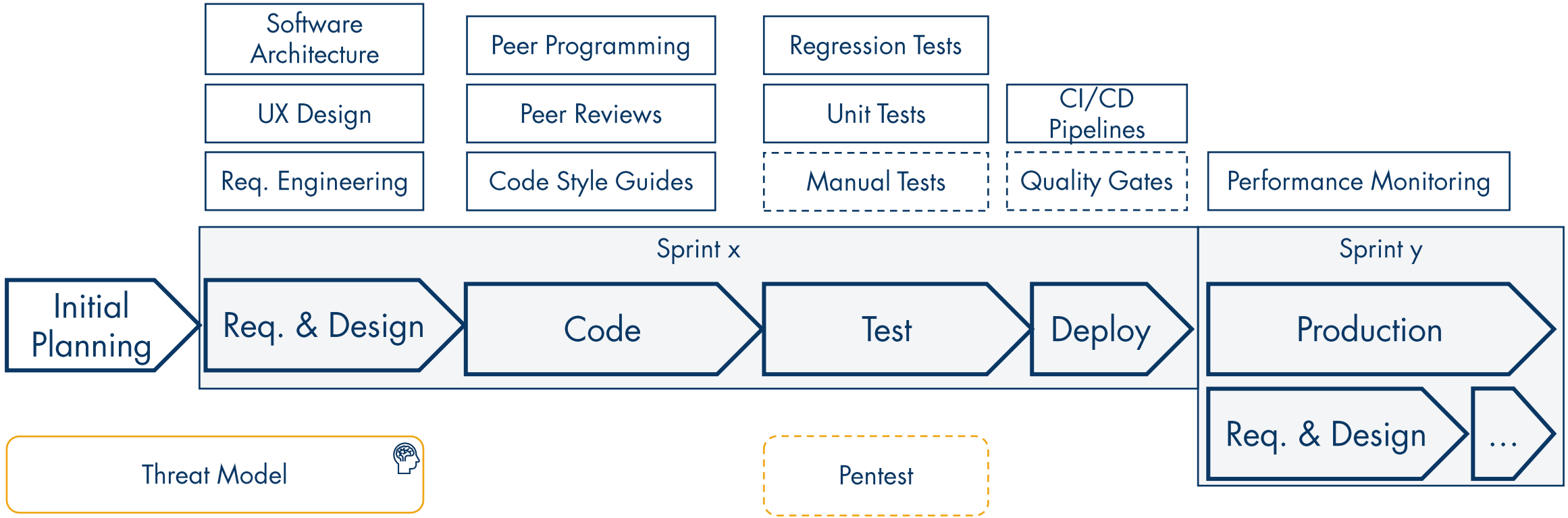


Multiple pentests in every sprint?



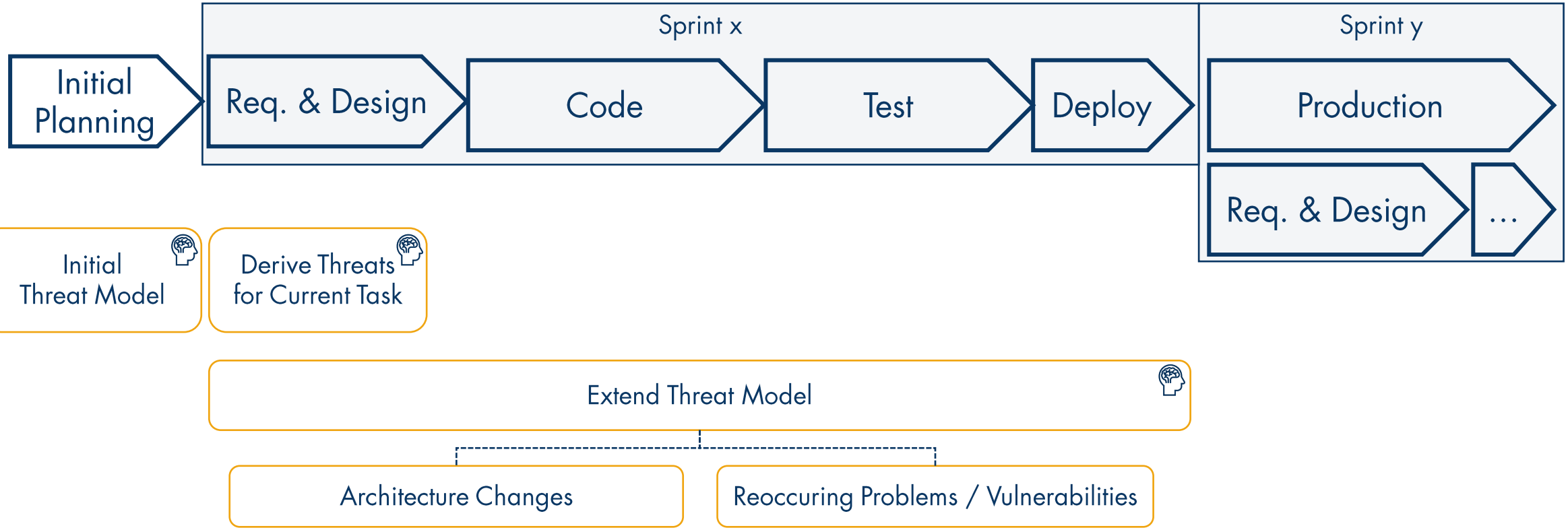






Threat Modeling

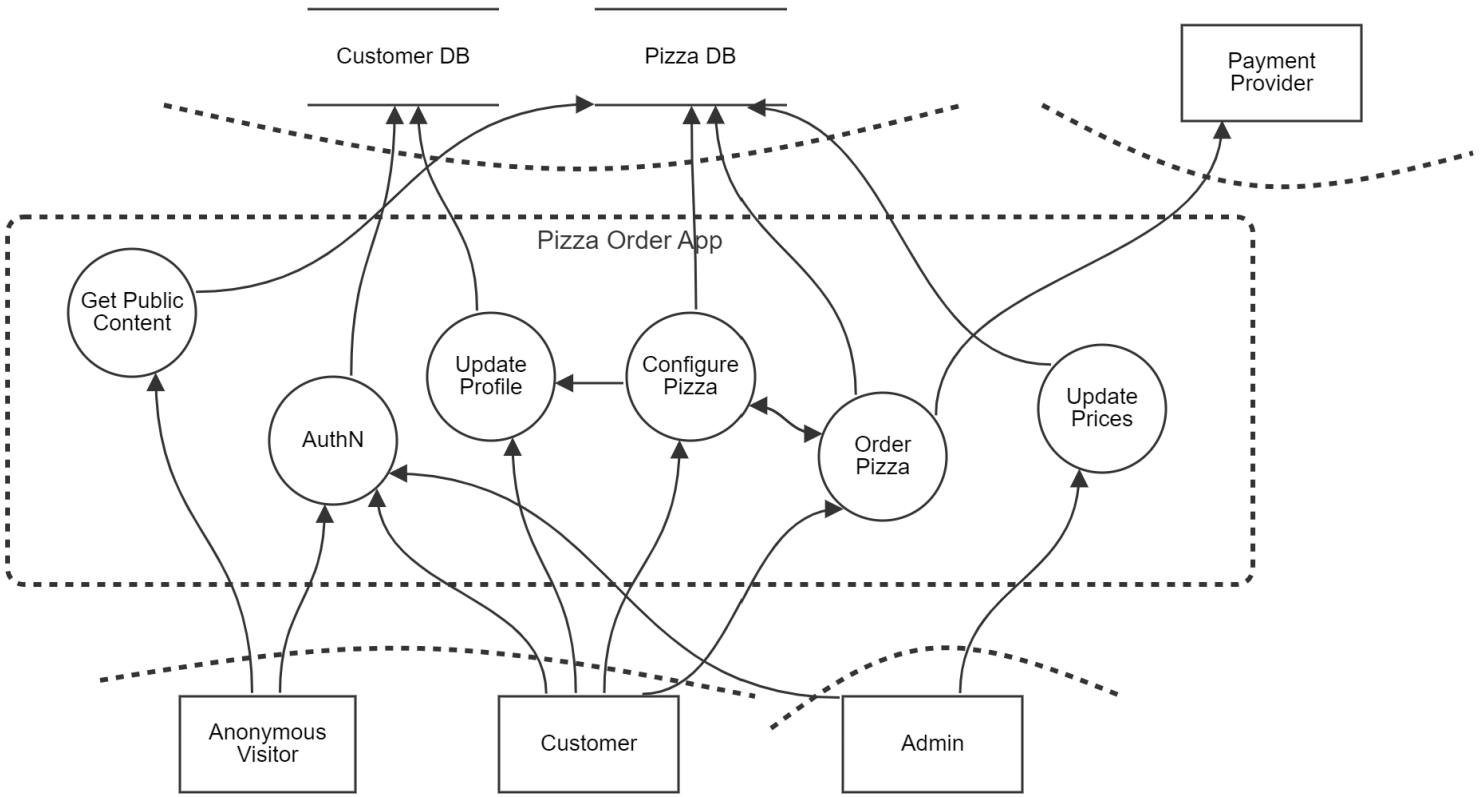
Identify things that can go wrong in your system and prevent them



Threat Modeling

1 What are we building?

- Identify your sensitive data
- Create a diagram together



little example:

- We're creating a pizza order system
- Customers can save
  - their address
  - their personal pizza configurations
  - their credit card
- Payment is done via an external payment-provider

Sensitive Data
credit card data
usernames
passwords
pizza prices
pizza configurations
....

Threat Modeling

1 What are we building?

- Identify your sensitive data
- Create a diagram together

2 What can go wrong?

- Brainstorm Worst-Case scenarios
- Use STRIDE as a checklist for major categories

Worst Case Szenarios	
Spoofing	Login as another customer and order a free pizza
Tampering	Competitor increases pizza prices; Customer modifies pizza prices to 0 €
Repudiation	Somebody orders a pizza and afterwards says he didn't
Information Disclosure	Credit Card data gets leaked
Denial of Service	Nobody can order pizza
Elevation of Privileges	Somebody can get admin privileges

Threat Modeling

1 What are we building?

- Identify your sensitive data
- Create a diagram together

2 What can go wrong?

- Brainstorm Worst-Case scenarios
- Use STRIDE as a checklist for major categories

3 What can we do about it?

- Use OWASP ASVS to derive requirements

Application Security Verification Standard 4.0

2.1.1	Verify that user set **passwords** are at least **12 characters** in length.
2.2.1	Verify that **anti-automation controls** are effective […] controls include **blocking the most common breached passwords, soft lockouts**, rate limiting, CAPTCHA, ever increasing delays between attempts […]
2.2.4	Verify impersonation resistance against phishing, such as the use of **multi-factor authentication** […]
9.1.1	Verify that **secured TLS** is used for all client connectivity, and does not fall back to insecure or unencrypted protocols.
…	…

Using ChatGPT as a Threat Model „Expert“

- Give it a lot of details for your initial threat model
- Shape the prompt according to best practices
  - e.g.: https://danielmiessler.com/p/response-shaping-how-to-move-from-ai-prompts-to-ai-whispering/
    - Tell the system who to behave as
    - Tell the system what format it produces
    - Give it the main task you want done
    - …
- Explain your current task for the ongoing threat model

The initial Threat Model

You

Hi,
please act as a cyber security professional who is a master in threat modeling.
Your job is to create a threat model for the following system:

We plan to build an online pizza order system. Customers can save their address, their personal preferred pizza configurations and their credit card data.

Of course the system also needs to save all the different available pizzas and their current prices. And for user logins it will save usernames and passwords.
There are three types of users:
1) anonymous users, who can only view public content like our pizza menu
2) authenticated customers, who can update their profile, configure pizzas and order these pizzas.
3) authenticated admins, who can update the pizza prizes

The whole system runs in containers in AWS. Data is stored in a PostgreSQL database.
The frontend is a React SPA which communicates with a JSON REST API written in Python.
The authentication and session management is done via JWTs

The payment of the pizza orders is done via an external payment provider.
The communication with the external payment provider will be done via a REST API.

Please create a threat model for this system.
It should contain 4 chapters:

1) Management Summary
This should be a short management summary to give an overview of the system.

2) Worst-Case Szenarios
This chapter should list a few worst-case szenarios the system faces and their potential business impact. There should be at least one realistic worst-case szenario for each STRIDE category.

3) Identified Threats
This is the main chapter which contains all possible identified threats and potential vulnerabilities (at least 10) in an extensive markdown table. The table should have the following columns: ID, Title, Threat Scenario, Potential Vulnerability, Impact, Risk, Countermeasures, ASVS-Requirements.
The column ASVS-Requirements should contain the IDs of the OWASP ASVS-Requirements relevant to the respective finding.

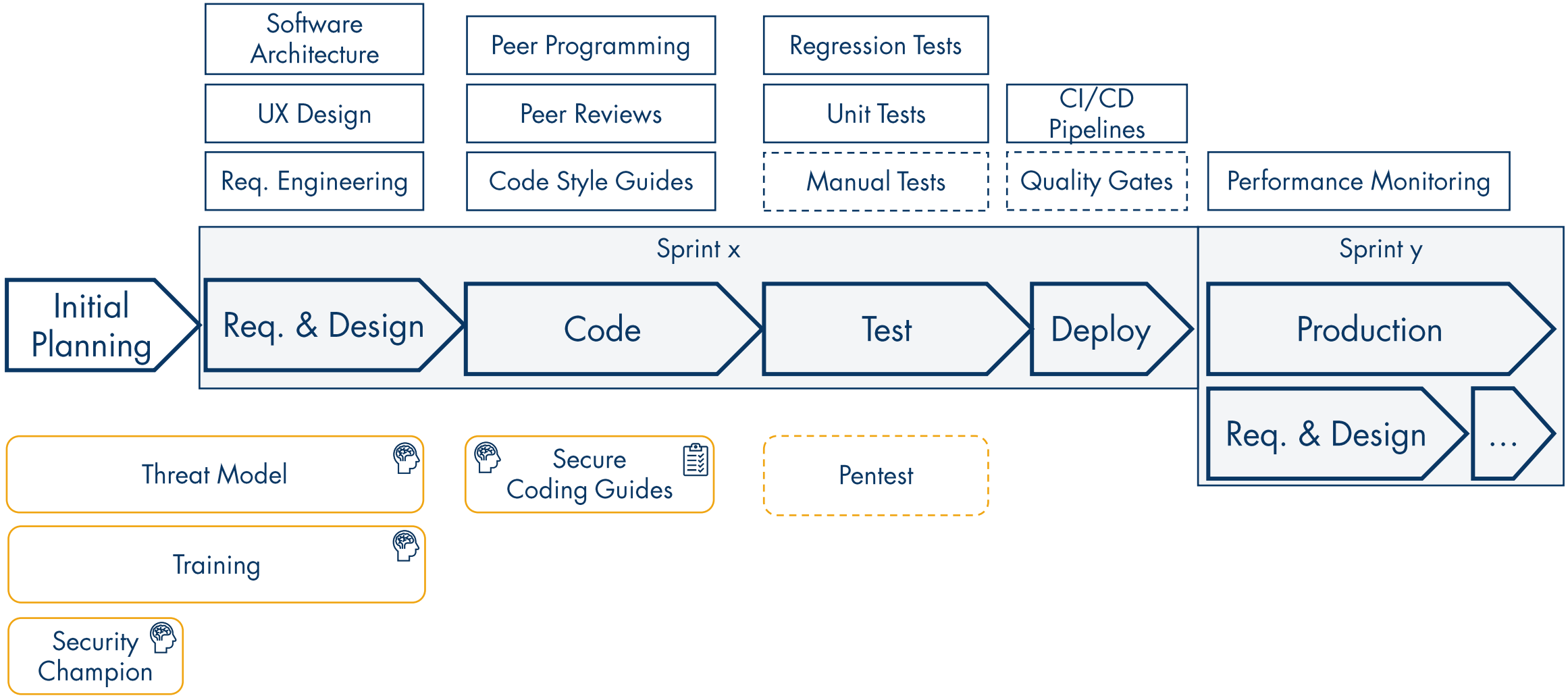
4) Countermeasure details
This chapter contains a detailed description of the countermeasures. It explains the benefits the countermeasure brings. Wherever possible please inlcude code or configuration examples, but also describe what exactly these examples are doing.

The ongoing Threat Model

DA

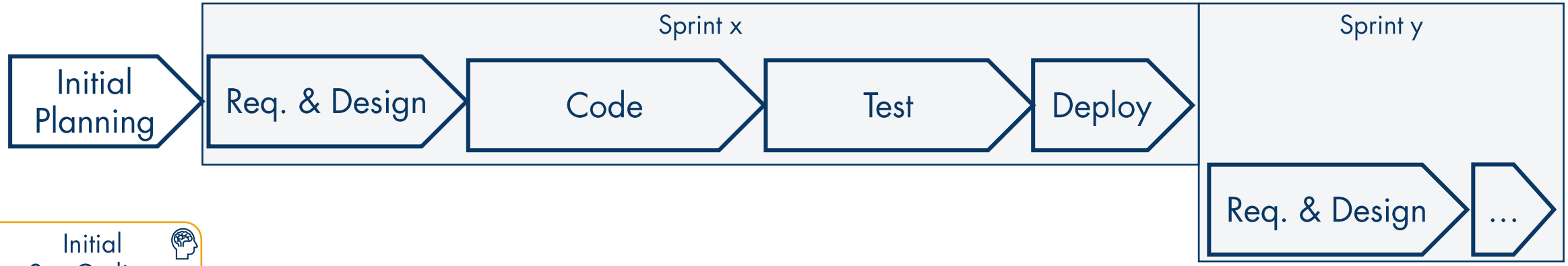
You

We would like to extend our system for a function to share pizza configurations with other users. So it should be possible for a user to create a link to a specific pizza configuration he defined, share this link with a friend and this friend should be able to add this configuration to his private profile.
Do you see any threats we should consider here?



Secure Coding Guidelines

Checklist to ensure secure coding best practices

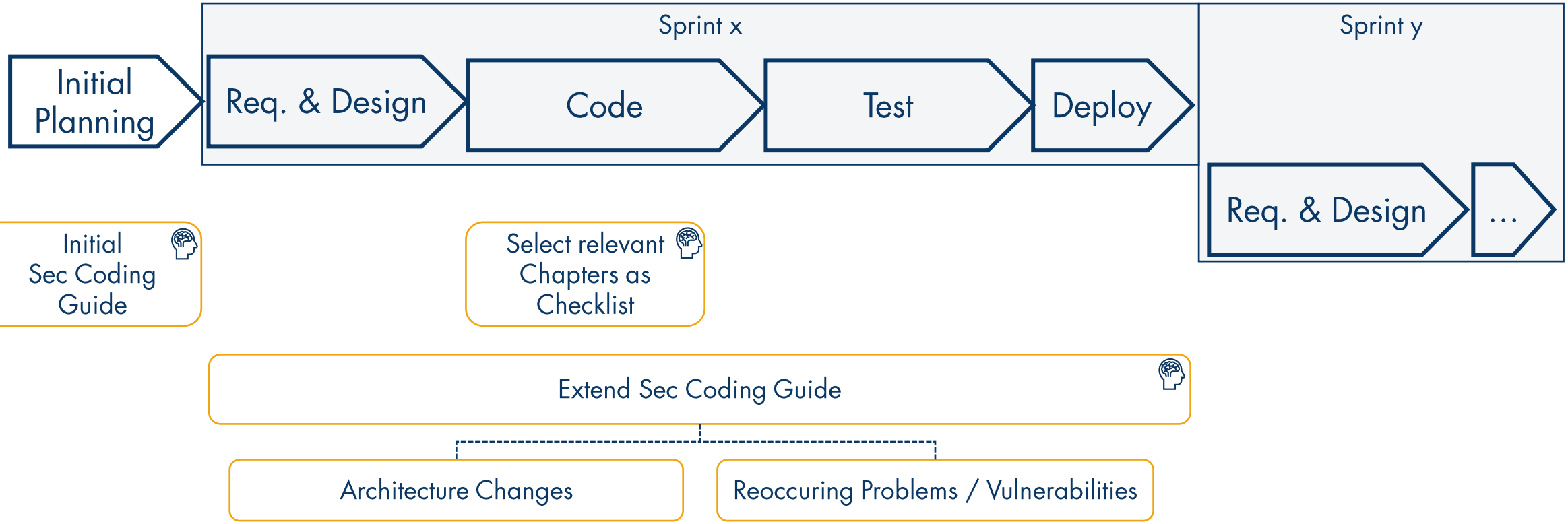


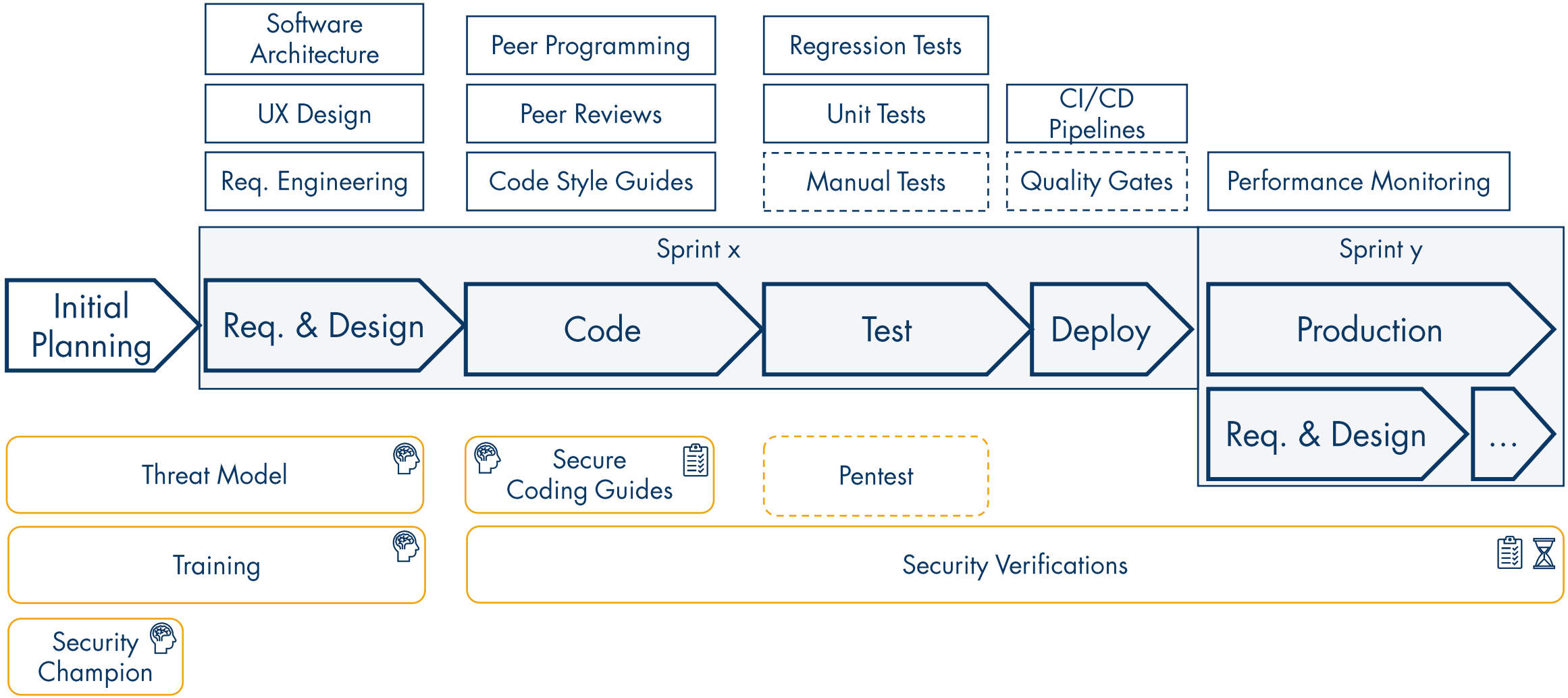
Initial Sec Coding Guide

- Derive from Company- or Generic Standard
- Get Expert Help
- Use Threat Model as Input

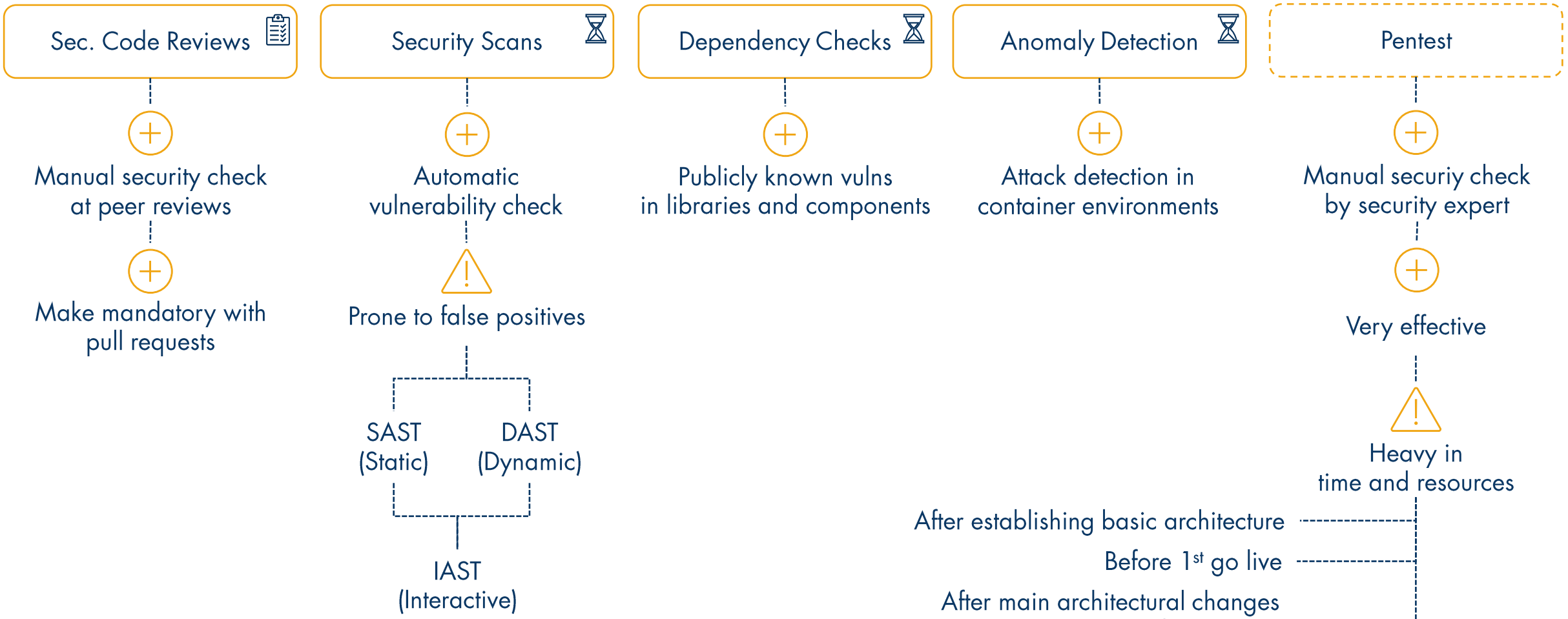
Secure Coding Guidelines

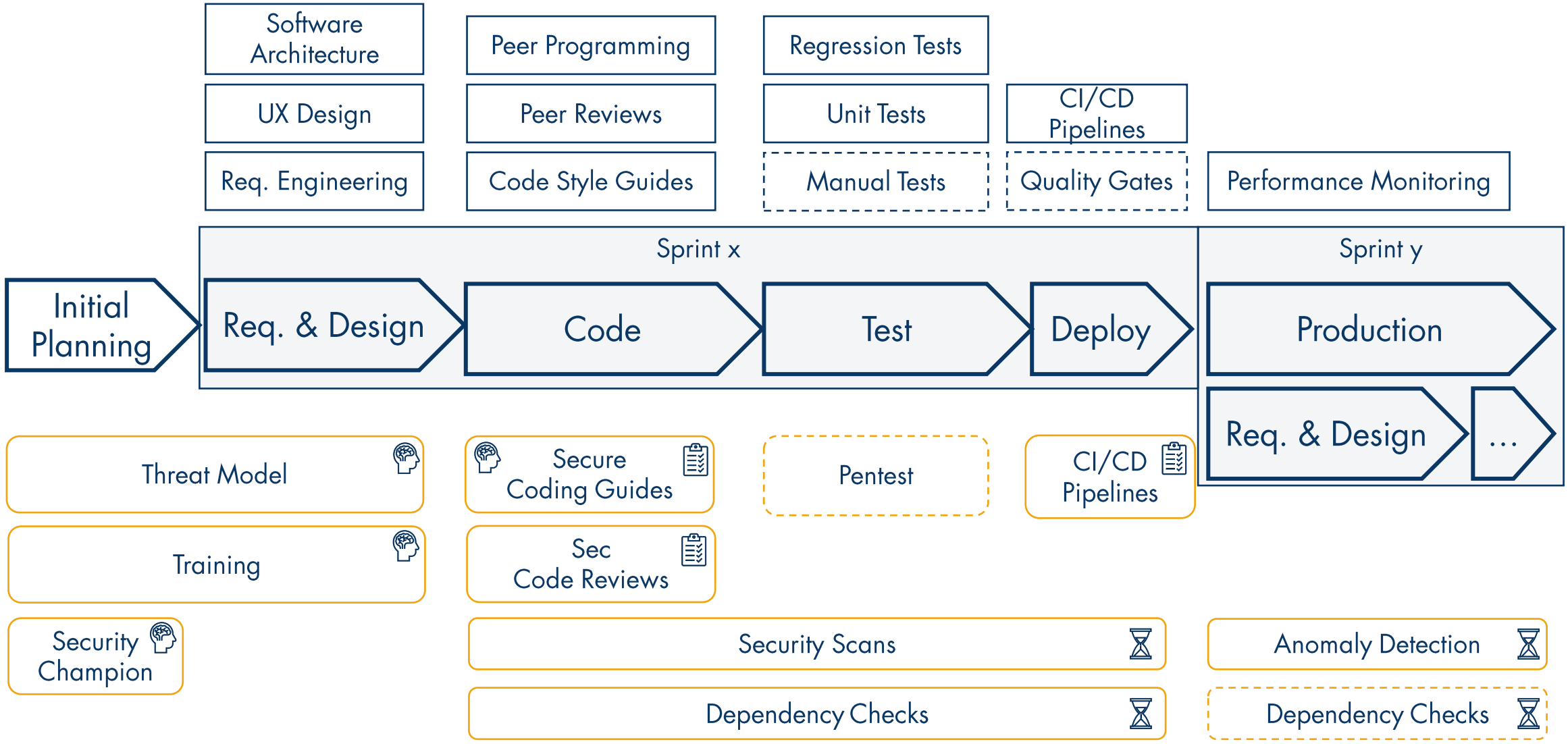
Checklist to ensure secure coding best practices

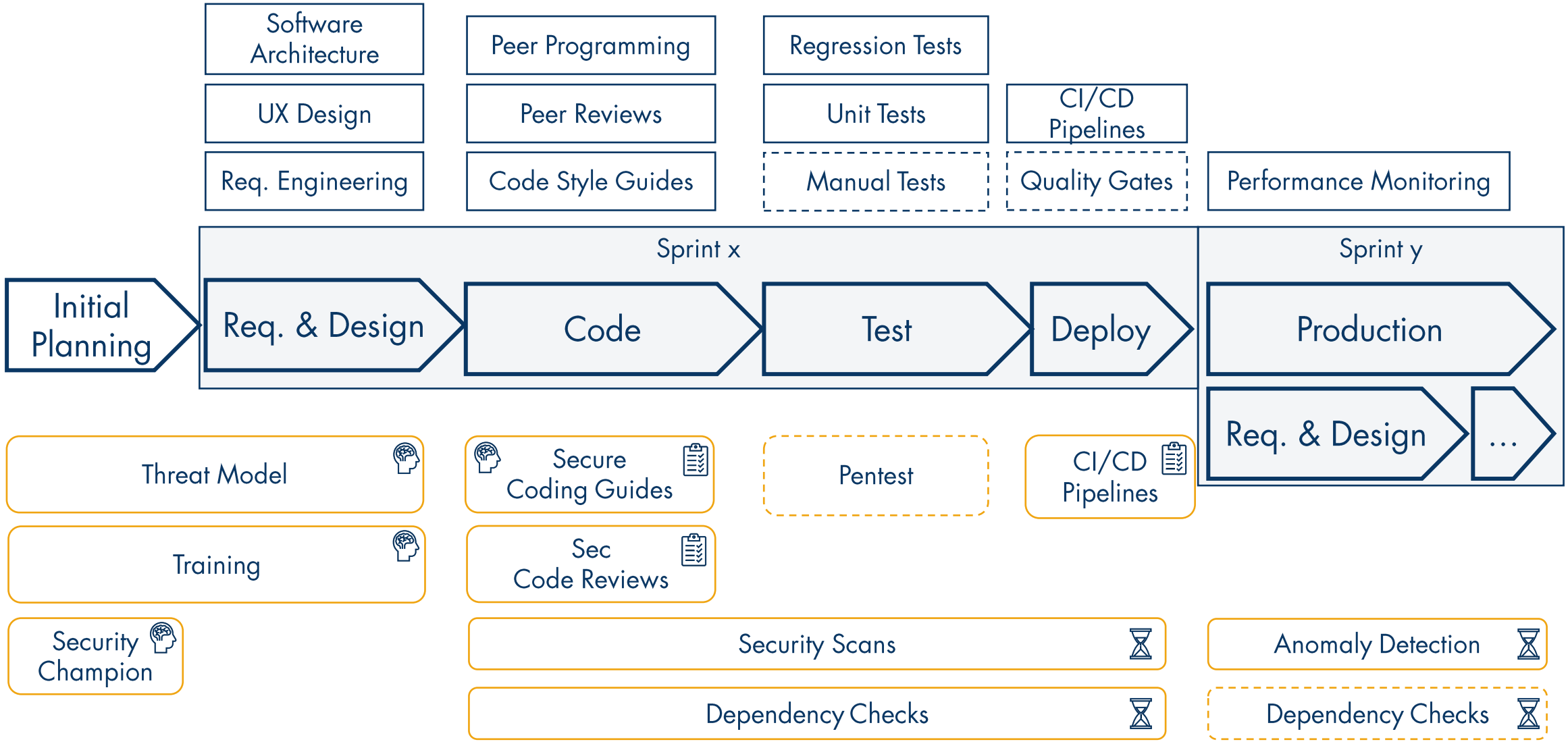




Security Verifications







Key Messages

1. **Security is more important than ever**
   We also need to consider it in short release cycles

2. **Automate (most) security tests and make them mandatory by pipelines**
   Smart combination of tools and manual pentests is most efficient

3. **Start threat modeling and utilize AI**
   A little brainstorming for worst-case szenarios already gives you a good starting point

4. **Use secure coding guidelines as checklists**
   Its hard to remember everything – checklists are great!

Resources

- Diagrams
  - Data Flow Diagrams (DFD): https://docs.microsoft.com/en-us/learn/modules/tm-introduction-to-threat-modeling/2-step-1-design-phase
  - C4 Model: https://c4model.com/
- Secure Design
  - 8 Security Design Principles: https://www.cs.virginia.edu/~evans/cs551/saltzer/
  - IEEE CSD: https://cybersecurity.ieee.org/center-for-secure-design/
- Threat Modeling Methods
  - STRIDE: https://www.microsoft.com/security/blog/2007/09/11/stride-chart/
  - OCTAVE: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=13473
  - TRIKE: http://www.octotrike.org/
  - PASTA: https://owasp.org/www-pdf-archive/AppSecEU2012_PASTA.pdf
  - LINDDUN: https://www.linddun.org
  - ATASM: https://published-prd.lanyonevents.com/published/rsaus18/sessionsFiles/9225/LAB3-R02_Threat-Models-Into-The-Deep.pdf
  - Attack Trees: https://www.schneier.com/academic/archives/1999/12/attack_trees.html
- Starting Points for Secure Coding Guides
  - OWASP ASVS: https://owasp.org/www-project-application-security-verification-standard/
  - OWASP Top 10: https://owasp.org/www-project-top-ten/
- AI Prompt Shaping
  - https://danielmiessler.com/p/response-shaping-how-to-move-from-ai-prompts-to-ai-whispering/

condignum

Daniel Schwarz
Senior Security Architect
daniel.schwarz@condignum.com