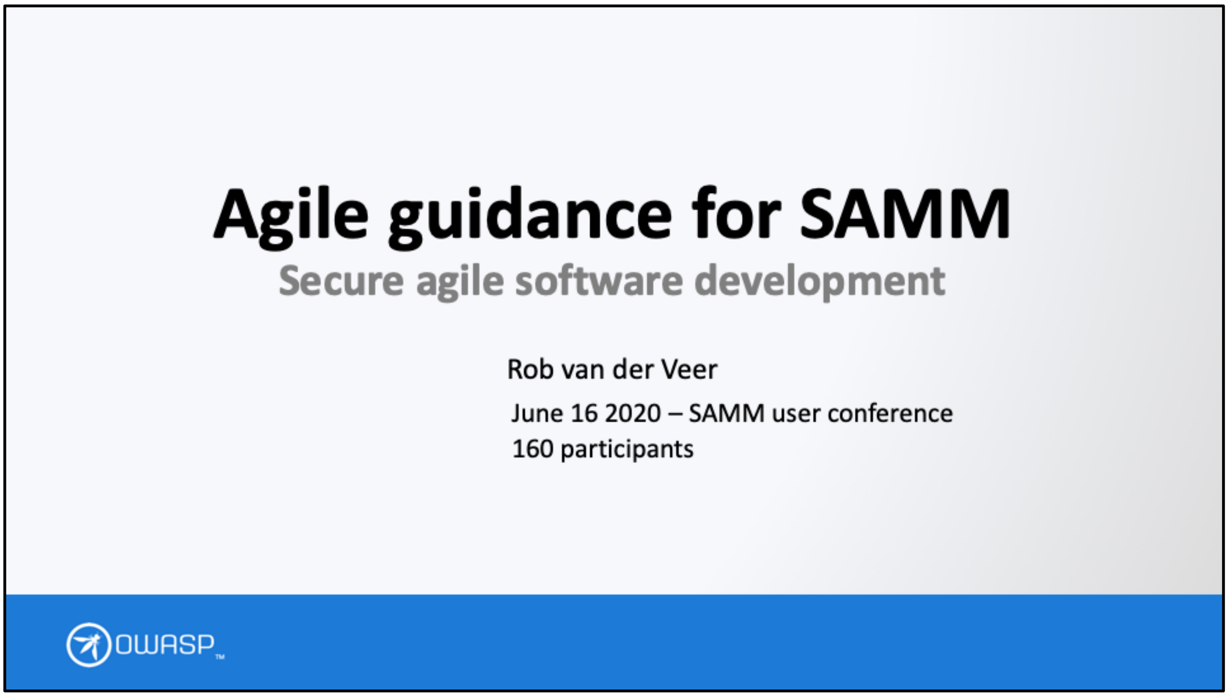


Agile guidance for SAMM

Secure agile software development

Rob van der Veer

June 16 2020 – SAMM user conference

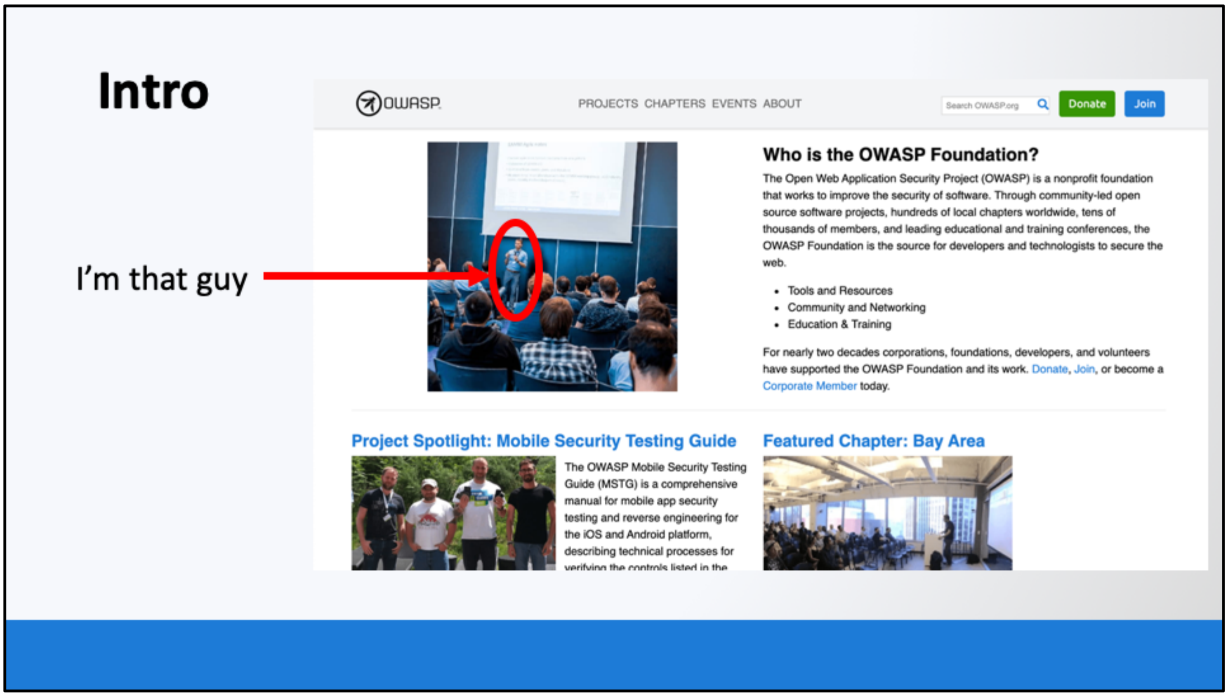
160 participants

Hello everybody. Wow, what an opportunity: a round table with 160 people! Let's get started right away.

In this session I would like to discuss the work we've done for SAMM on secure software development specifically for Agile.

My goal is to share knowledge. I will be talking about the guidance but I also invite you, to share your challenges, your best practices, your lessons learned, so that we can learn from eachother. There's a good bunch of people joining today, so let's benefit from that wisdom of the crowd. Because I definitely am not claiming to have all the answers So, please, during the session,use the chat to pose questions, comments and to respond to the questions that I will have for you. Nessim will be so kind to assist me with processing your input.

But let me first introduce myself.



Intro

I'm that guy

Who is the OWASP Foundation?

The Open Web Application Security Project (OWASP) is a nonprofit foundation that works to improve the security of software. Through community-led open source software projects, hundreds of local chapters worldwide, tens of thousands of members, and leading educational and training conferences, the OWASP Foundation is the source for developers and technologists to secure the web.

- Tools and Resources
- Community and Networking
- Education & Training

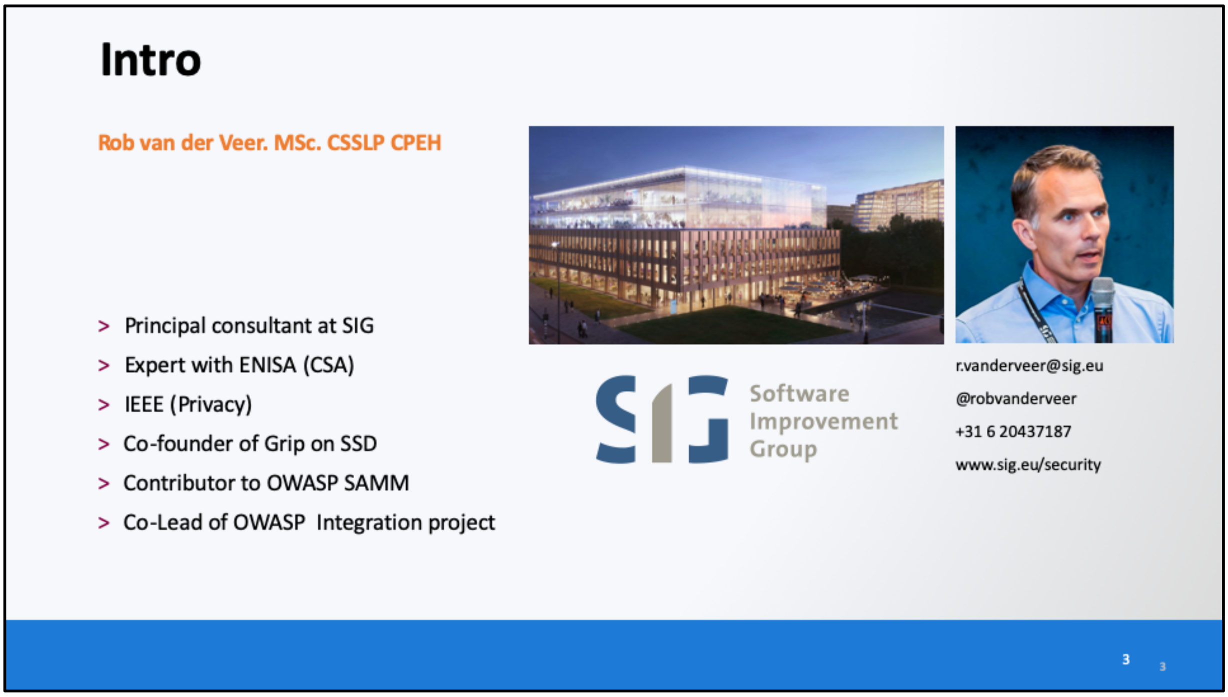
For nearly two decades corporations, foundations, developers, and volunteers have supported the OWASP Foundation and its work. Donate, Join, or become a Corporate Member today.

Project Spotlight: Mobile Security Testing Guide

The OWASP Mobile Security Testing Guide (MSTG) is a comprehensive manual for mobile app security testing and reverse engineering for the iOS and Android platform, describing technical processes for verifying the controls listed in the

Featured Chapter: Bay Area

I have been at the frontpage of OWASP for a while now. Not because I'm important, but by pure coincidence. It's a picture of me presenting at Global Appsec Amsterdam.



Intro

Rob van der Veer. MSc. CSSLP CPEH

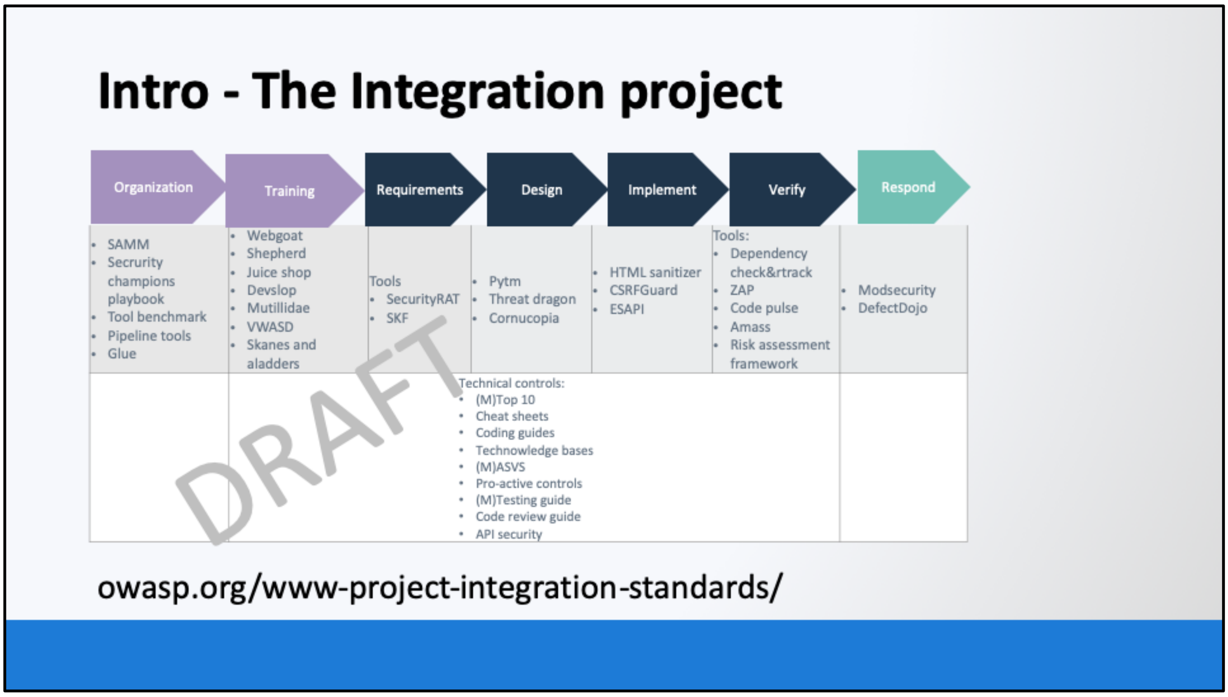
- Principal consultant at SIG
- Expert with ENISA (CSA)
- IEEE (Privacy)
- Co-founder of Grip on SSD
- Contributor to OWASP SAMM
- Co-Lead of OWASP Integration project

Software
Improvement
Group

r.vanderveer@sig.eu
@robvanderveer
+31 6 20437187
www.sig.eu/security

SIG measures quality of software and guide organisations through shifting left, with offices in Amsterdam, New York, Antwerp, Copenhagen and Frankfurt.
ENISA : I worked on the recent report “Advancing software security in the EU”

You might have seen me or my team at one of the EU Appsec events. Let’s hope we can start doing that again in the future.



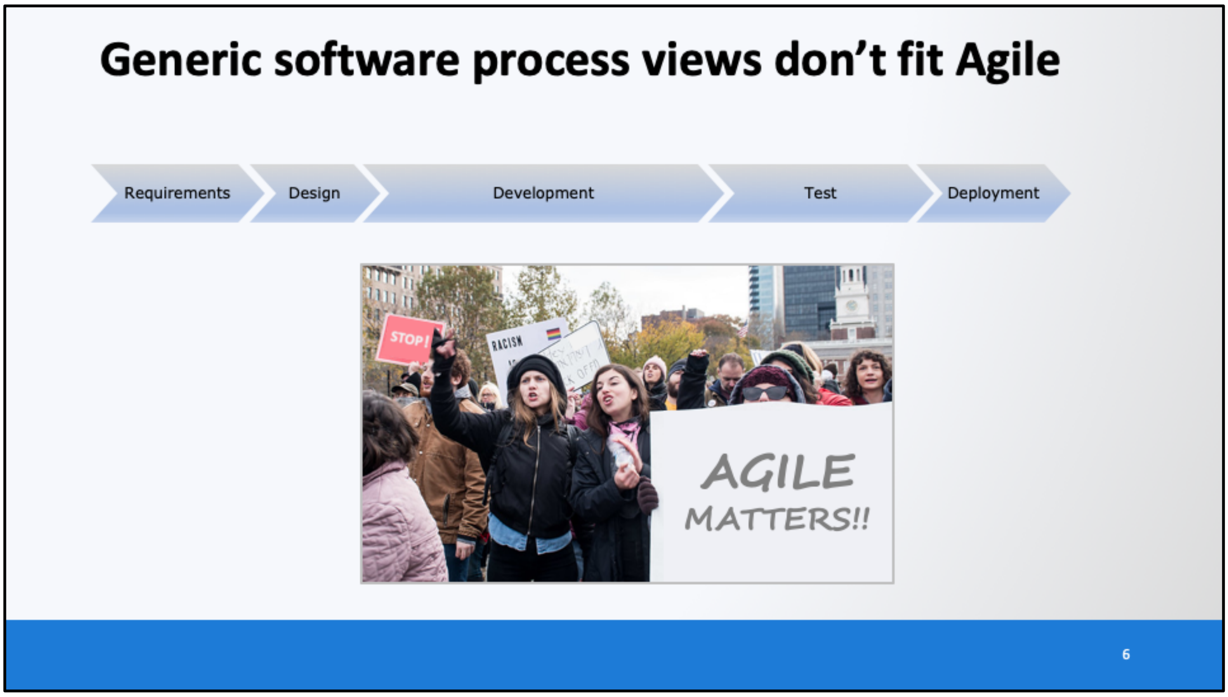
Intro - The Integration project

Organization	Training	Requirements	Design	Implement	Verify	Respond
• SAMM • Secrurity champions playbook • Tool benchmark • Pipeline tools • Glue	• Webgoat • Shepherd • Juice shop • Devslop • Mutillidae • VWASD • Skanes and aladders	Tools • SecurityRAT • SKF	• Pytm • Threat dragon • Cornucopia	• HTML sanitizer • CSRFGuard • ESAPI	Tools: • Dependency check&rtrack • ZAP • Code pulse • Amass • Risk assessment framework	• Modsecurity • DefectDojo
	Technical controls: • (M)Top 10 • Cheat sheets • Coding guides • Technowledge bases • (M)ASVS • Pro-active controls • (M)Testing guide • Code review guide • API security					

owasp.org/www-project-integration-standards/

At the Integration project I work with Spyros, Elie and the team on providing overview over OWASP projects and an initiative called Common Requirement Enumeration where we link standards inside and outside OWASP at the level of requirements/controls.

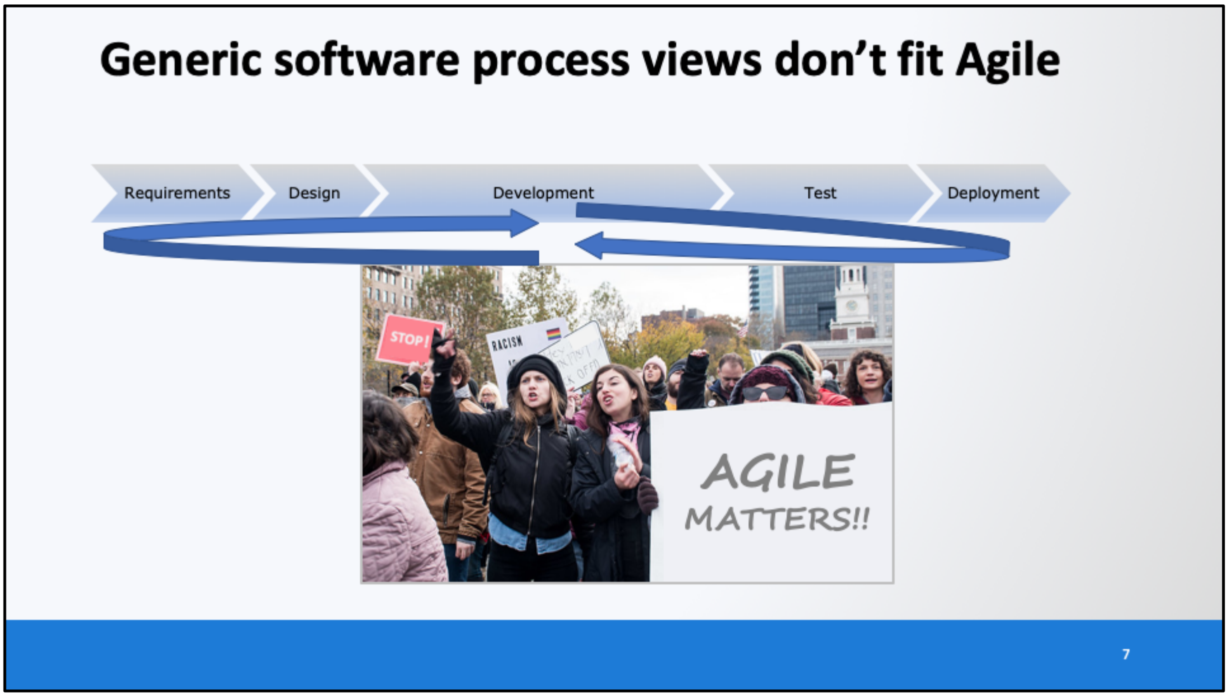
But. Enough about me. Let’s talk about Agile secure development.



Generic software process views don't fit Agile

Requirements → Design → Development → Test → Deployment

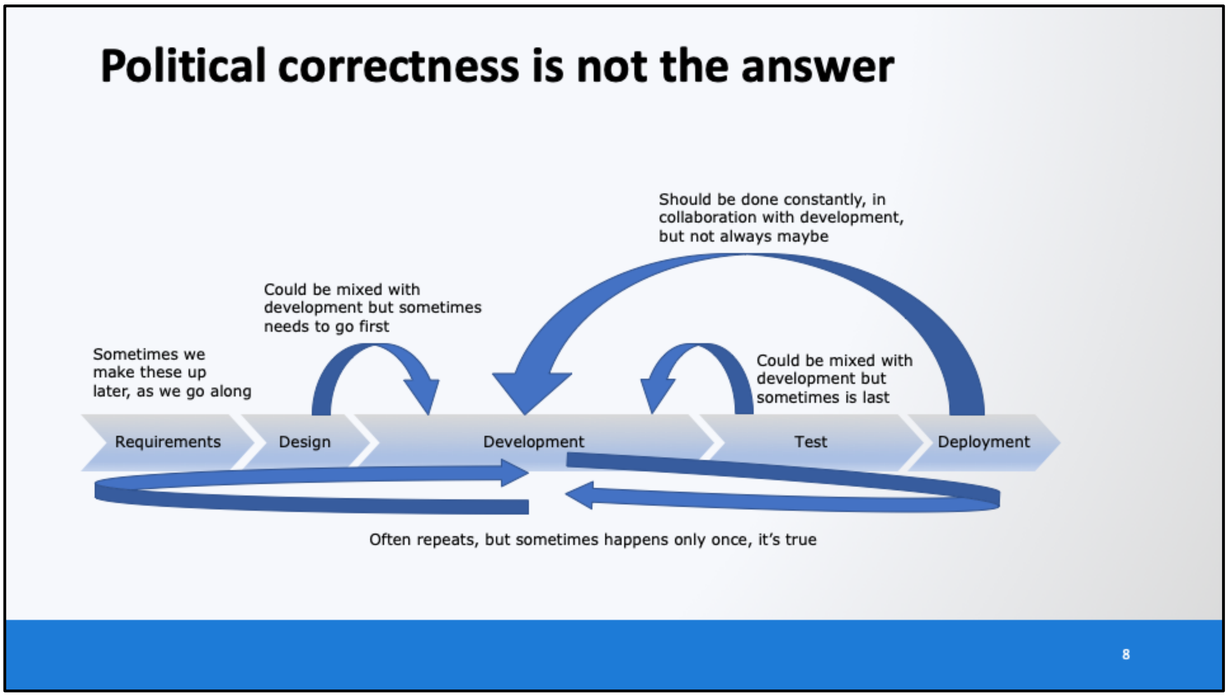
When I do a presentation on software development I typically would show this diagram.
And through the years I increasingly got complaints and people correcting me, saying that this is completely not agile, and thereby everything I say about software engineering will be incorrect..
So I often did this:



Generic software process views don't fit Agile

Requirements → Design → Development → Test → Deployment

And it does Agile more justice, but it doesn't provide the guidance that is needed. Even if I would draw a politically correct diagram, that would resonate with Devops, Waterfall, Agile and everybody:



Political correctness is not the answer

Should be done constantly, in collaboration with development, but not always maybe

Could be mixed with development but sometimes needs to go first

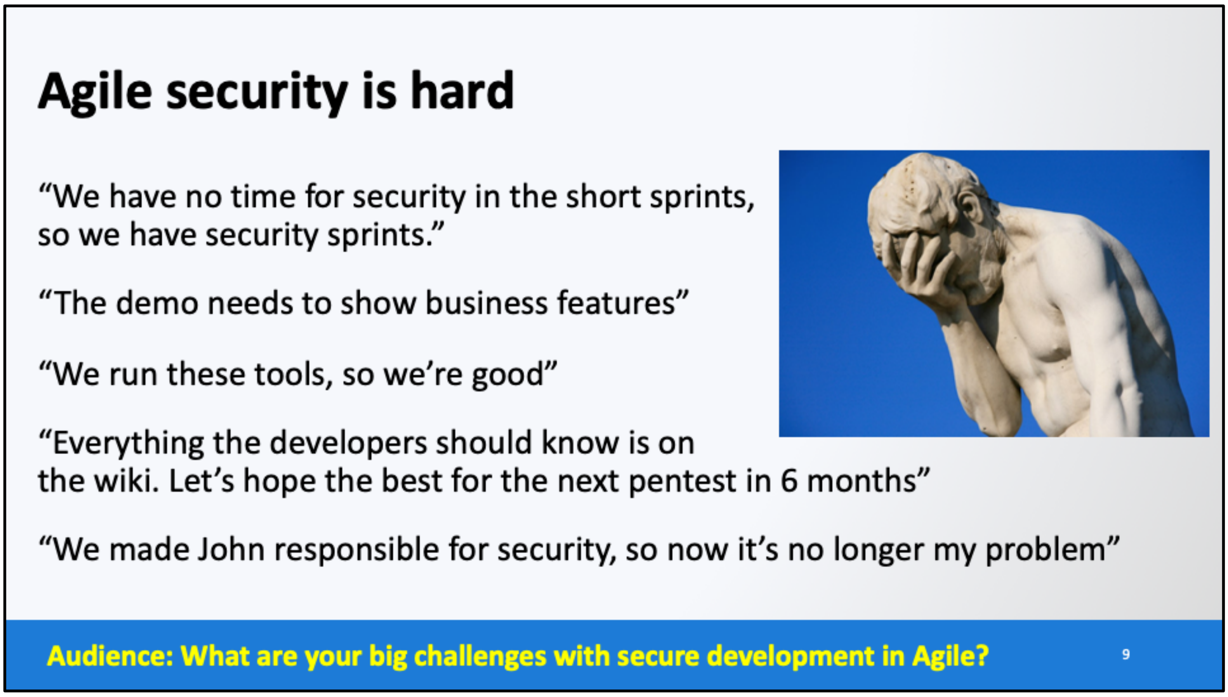
Sometimes we make these up later, as we go along

Could be mixed with development but sometimes is last

Often repeats, but sometimes happens only once, it's true

8

…. it would do justice to the nuances on a high level, but still wouldn't answer the many questions that Agile teams have. And there are a lot of questions, because:



Agile security is hard

"We have no time for security in the short sprints, so we have security sprints."

"The demo needs to show business features"

"We run these tools, so we're good"

"Everything the developers should know is on the wiki. Let's hope the best for the next pentest in 6 months"

"We made John responsible for security, so now it's no longer my problem"

Audience: What are your big challenges with secure development in Agile?

Agile security is hard, as illustrated by these quotes from some of the organizations I worked with.

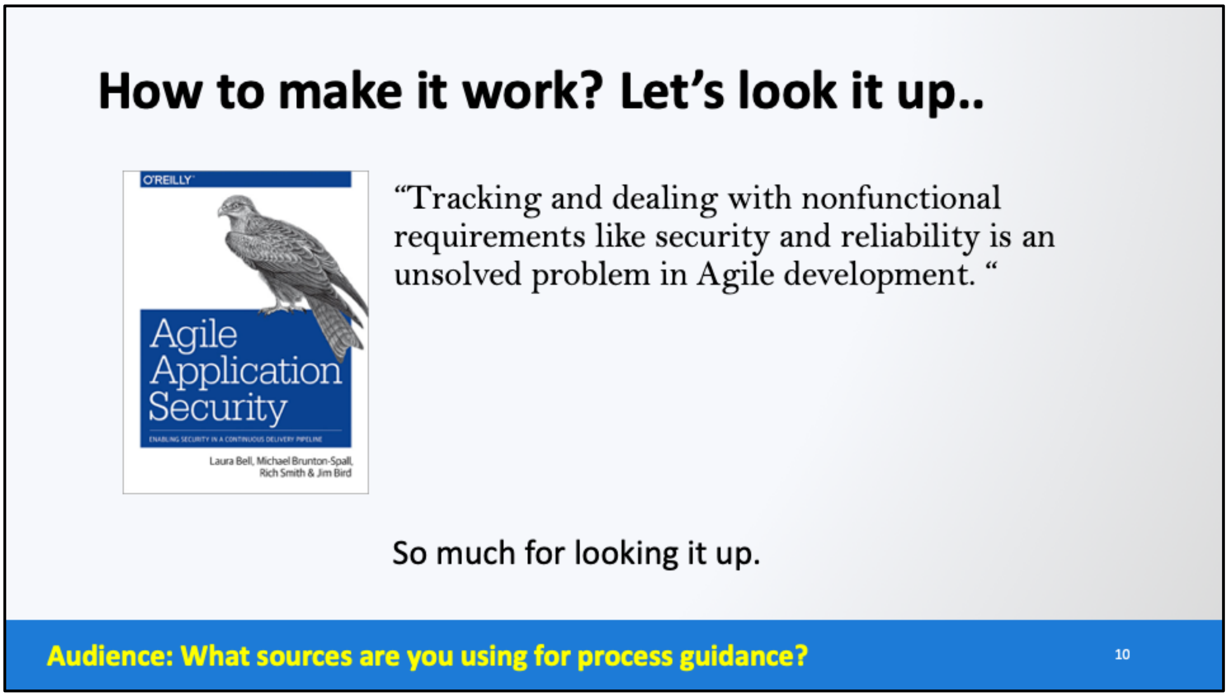
*No time: So, traditionally, quality assurance has been a carefully designed single phase in software development, with its own bureaucray. Of course there's no time for that in sprints. And of course you can focus some sprints on security features, but also the non-functional security should be built in continuously. How else are you going to deliver working software? 'Working' doesn't mean that it has the features that it needs to have. It needs to work in the real world. So the key is to find a way to make security fit.

*Demo: I often see an overfocus on features caused by the idea that sprint demos are about functions only. It's important for the business to make sure security is regarded as a business value as well.

*Tools: using tools is a good practice but it's important to understand the limitations of these tools.

*Wiki: gathering requirements and guidelines is a good thing, but that’s not where it ends. You need an effective way for the team to get access to the relevant requirements based on the type of task they are working on. Having them on a wiki or in a document will not suffice.

There appears to be a strong need in the industry for guidance on how to make secure software development work in an Agile environment. So, how can we make it work?



How to make it work? Let’s look it up..

O'REILLY
Agile Application Security
ENABLING SECURITY IN A CONTINUOUS DELIVERY PIPELINE
Laura Bell, Michael Brunton-Spall, Rich Smith & Jim Bird

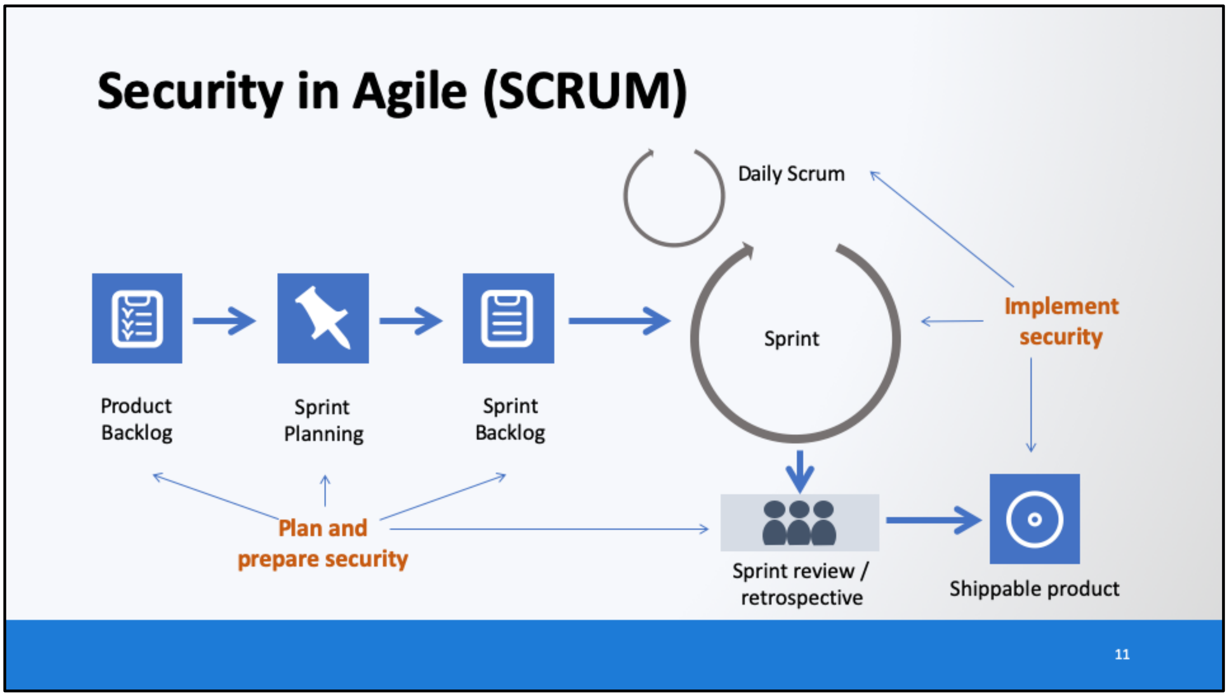
“Tracking and dealing with nonfunctional requirements like security and reliability is an unsolved problem in Agile development. “

So much for looking it up.

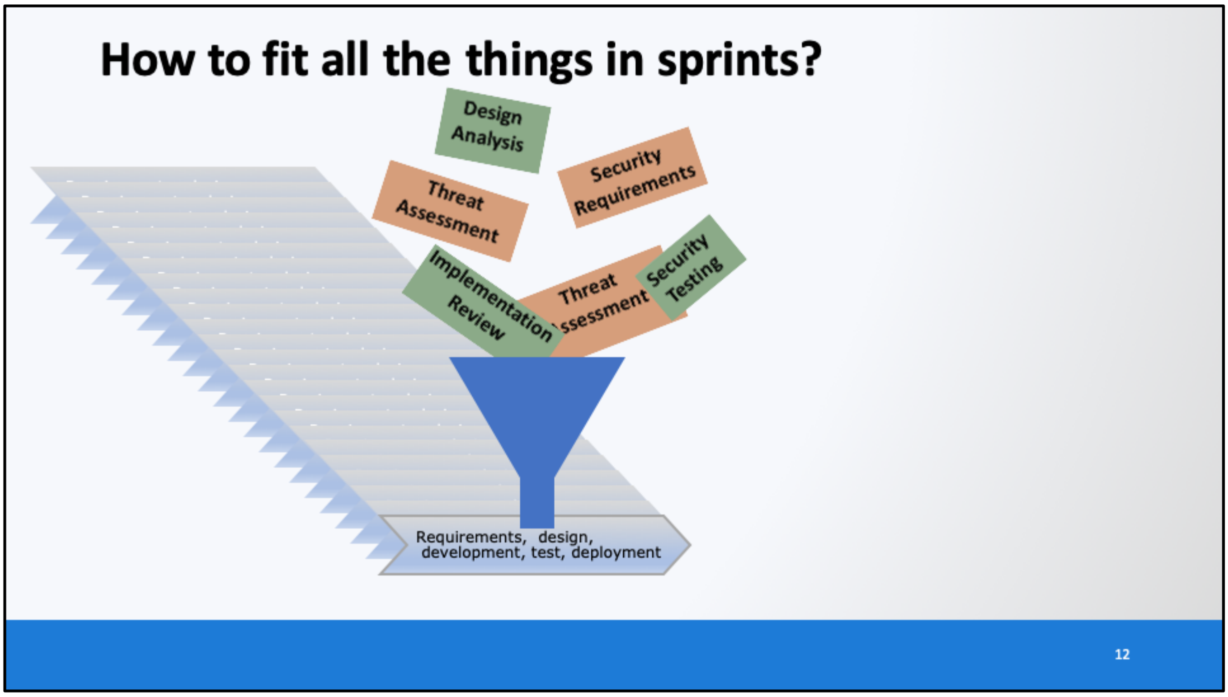
Audience: What sources are you using for process guidance?

10

Let’s have a quick look at what we are dealing with:



Security in Agile (SCRUM)



How to fit all the things in sprints?

Design Analysis

Threat Assessment

Security Requirements

Implementation Review

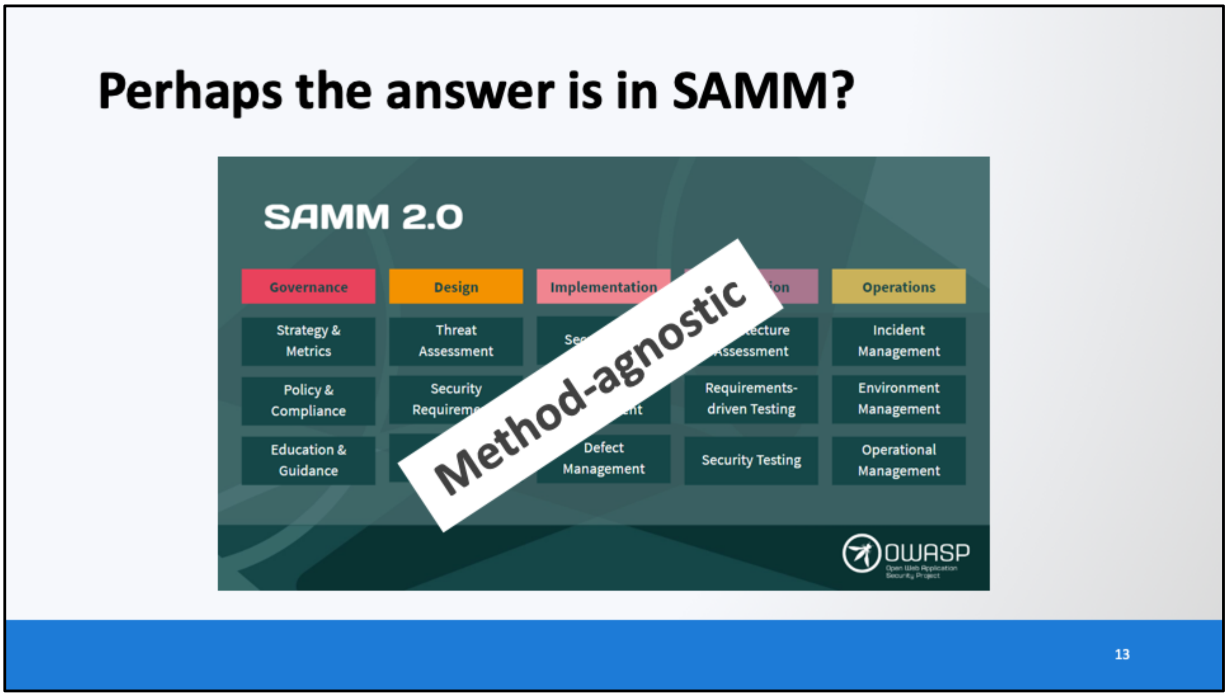
Threat Assessment

Security Testing

Requirements, design, development, test, deployment

12

You do many short iterations and every iteration you want to deliver software that is ready, with security built in.
So how do you squeeze all the necessary activities into a sprint: requirement selection, threat modelling, verification? What do you do with stories, with abuse stories and with the Definition Of Done? How do you get rid of the bureaucracy that is sometimes involved in how security teams and developers co-operate?



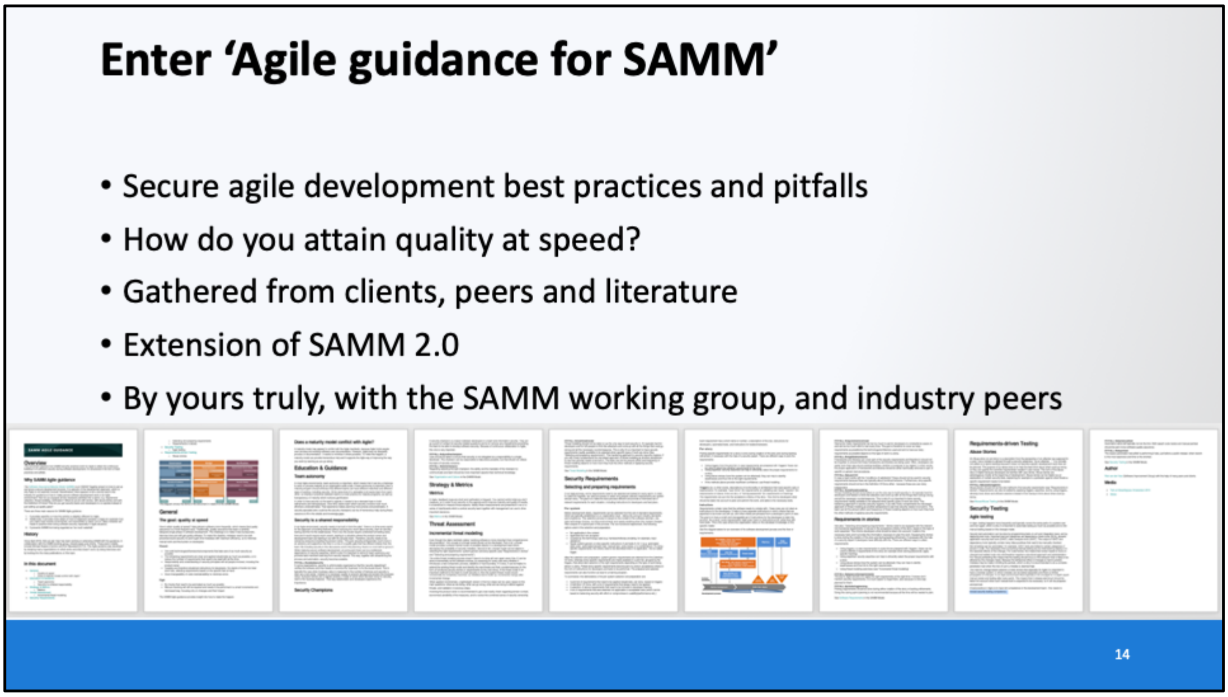
Perhaps the answer is in SAMM?

SAMM 2.0

Governance	Design	Implementation	ion	Operations
Strategy & Metrics	Threat Assessment	Se	ecture Assessment	Incident Management
Policy & Compliance	Security Requireme	nt	Requirements-driven Testing	Environment Management
Education & Guidance		Defect Management	Security Testing	Operational Management

13

So, SAMM discusses business functions and practices, but at the same time it aims to be agnostic of the type of development approach, which is why Agile was not covered. This is a good decision to make the model elegant.
And therefore:



Enter ‘Agile guidance for SAMM’

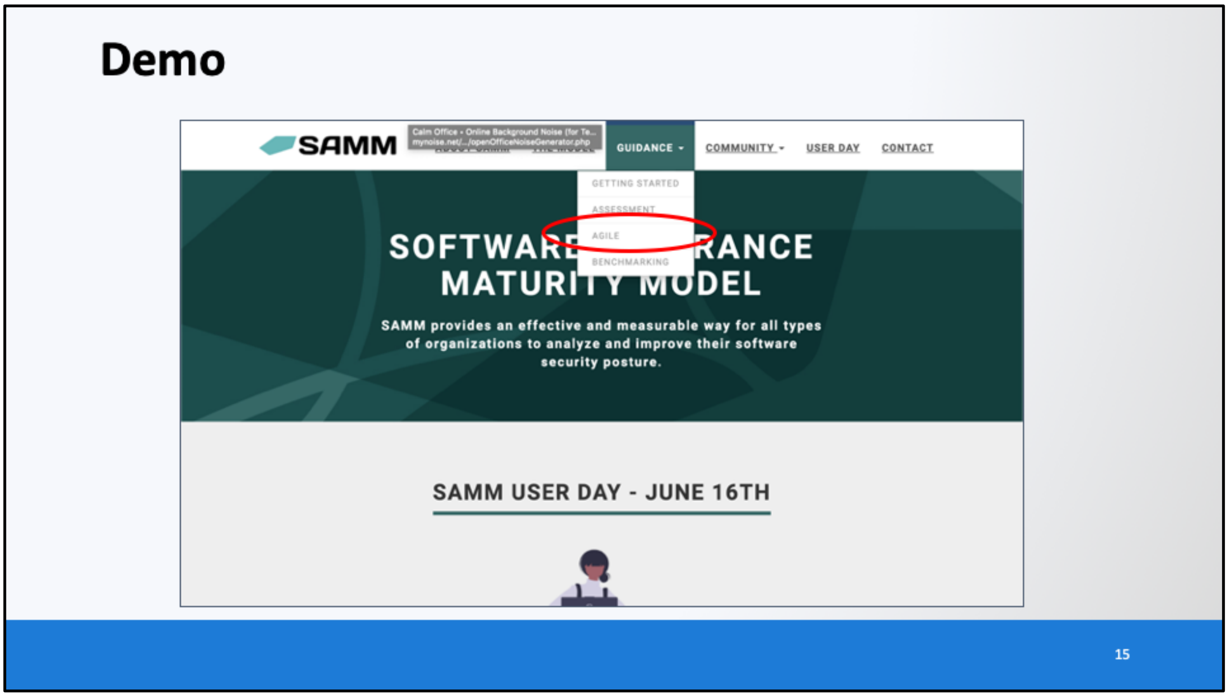
- Secure agile development best practices and pitfalls
- How do you attain quality at speed?
- Gathered from clients, peers and literature
- Extension of SAMM 2.0
- By yours truly, with the SAMM working group, and industry peers

14

Agile guidance explains how SAMM practices work for Agile, in the form of best practices and pitfalls.
Since April 2018, I have been working on this project, in collaboration with the SAMM working group, clients and industry peers (eg. Michael Kuipers from Centric and Eric Nieuwland from Ictu). We did this based on our own experiences, by studying many organisations on what works and what doesn't work, by doing observations, interviews and by looking into the many publications on this topic.

This got initated because I was working on a Dutch government guideline on Agile security (CIP’s Agile Security Management) that was unknowingly repeating the work done withing SAMM and wanted to add agile best practices on top. I decided to go talk to the SAMM group and convinve the government that the guideline should refer to SAMM and that we would extend SAMM with agile guidelines that we would then also add and translate to the Dutch publication.

I will show you a detailed example of guidance later, and we’ll discuss the main principles behind the guidance. But first: what does it look like and what does it cover.



Demo

SAMM
Calm Office • Online Background Noise (for Te...
mynoise.net/.../openOfficeNoiseGenerator.php
GUIDANCE
COMMUNITY
USER DAY
CONTACT
GETTING STARTED
ASSESSMENT
AGILE
BENCHMARKING

SOFTWARE RANCE MATURITY MODEL

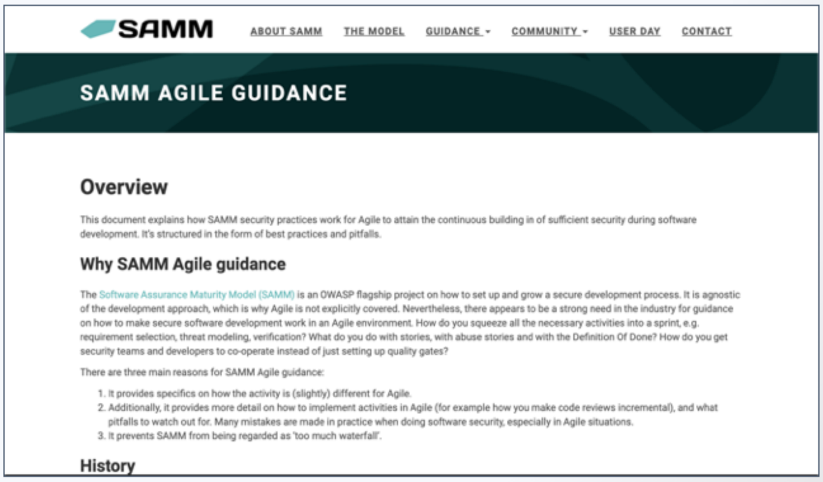
SAMM provides an effective and measurable way for all types of organizations to analyze and improve their software security posture.

SAMM USER DAY - JUNE 16TH

15

This integration in the SAMM website was released recently, with the great help of Patricia Duarte.

Demo - owaspsamm.org/guidance/agile/



SAMM AGILE GUIDANCE

Overview

This document explains how SAMM security practices work for Agile to attain the continuous building in of sufficient security during software development. It's structured in the form of best practices and pitfalls.

Why SAMM Agile guidance

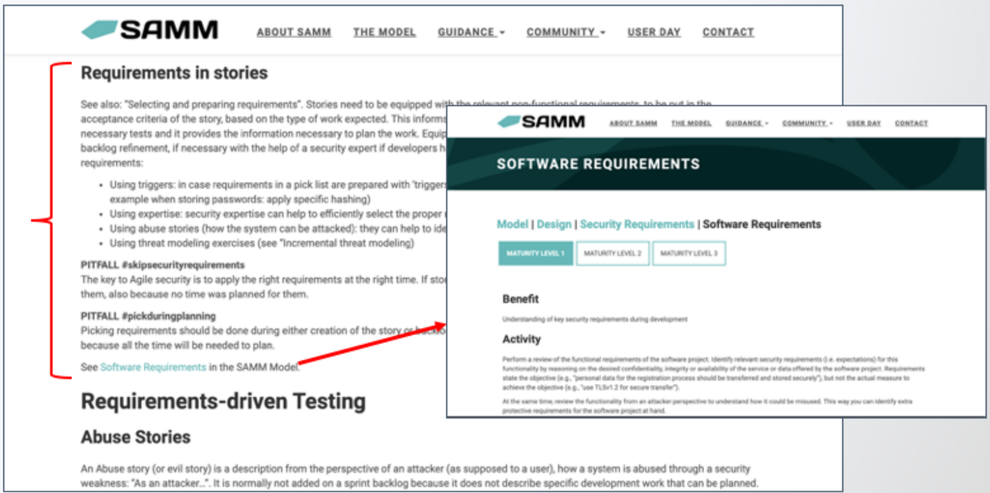
The Software Assurance Maturity Model (SAMM) is an OWASP flagship project on how to set up and grow a secure development process. It is agnostic of the development approach, which is why Agile is not explicitly covered. Nevertheless, there appears to be a strong need in the industry for guidance on how to make secure software development work in an Agile environment. How do you squeeze all the necessary activities into a sprint, e.g. requirement selection, threat modeling, verification? What do you do with stories, with abuse stories and with the Definition Of Done? How do you get security teams and developers to co-operate instead of just setting up quality gates?

There are three main reasons for SAMM Agile guidance:

1. It provides specifics on how the activity is (slightly) different for Agile.
2. Additionally, it provides more detail on how to implement activities in Agile (for example how you make code reviews incremental), and what pitfalls to watch out for. Many mistakes are made in practice when doing software security, especially in Agile situations.
3. It prevents SAMM from being regarded as 'too much waterfall'.

History

Demo



Requirements in stories

See also: "Selecting and preparing requirements". Stories need to be equipped with the relevant non-functional requirements, to be put in the acceptance criteria of the story, based on the type of work expected. This informs necessary tests and it provides the information necessary to plan the work. Equip backlog refinement, if necessary with the help of a security expert if developers h requirements:

- Using triggers: in case requirements in a pick list are prepared with 'triggers example when storing passwords: apply specific hashing)
- Using expertise: security expertise can help to efficiently select the proper
- Using abuse stories (how the system can be attacked): they can help to ide
- Using threat modeling exercises (see "Incremental threat modeling)

PITFALL #skipsecurityrequirements
The key to Agile security is to apply the right requirements at the right time. If sto them, also because no time was planned for them.

PITFALL #pickduringplanning
Picking requirements should be done during either creation of the story or backlo because all the time will be needed to plan.

See Software Requirements in the SAMM Model.

Requirements-driven Testing

Abuse Stories

An Abuse story (or evil story) is a description from the perspective of an attacker (as supposed to a user), how a system is abused through a security weakness: "As an attacker...". It is normally not added on a sprint backlog because it does not describe specific development work that can be planned.

SOFTWARE REQUIREMENTS

Model | Design | Security Requirements | Software Requirements

Benefit

Understanding of key security requirements during development

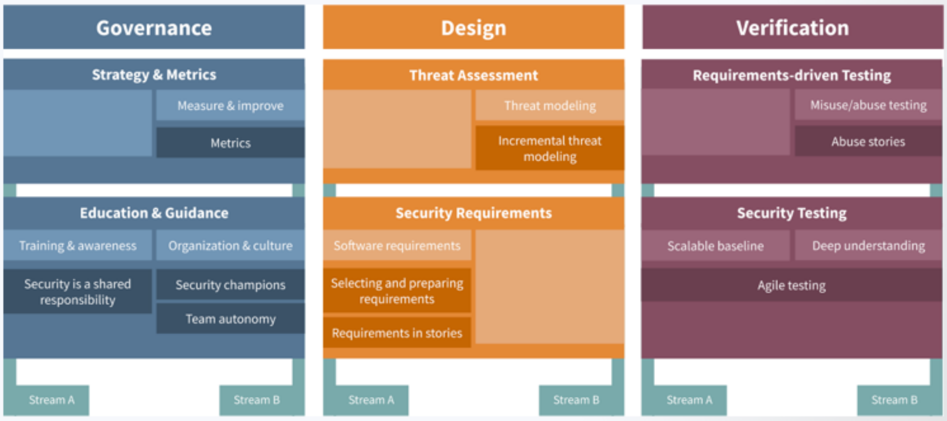
Activity

Perform a review of the functional requirements of the software project. Identify relevant security requirements (i.e. expectations) for this functionality by reasoning on the desired confidentiality, integrity or availability of the service or data offered by the software project. Requirements state the objective (e.g., "personal data for the registration process should be transferred and stored securely"), but not the actual measure to achieve the objective (e.g., "use TLSv1.2 for secure transfer").

At the same time, review the functionality from an attacker perspective to understand how it could be misused. This way you can identify extra protective requirements for the software project at hand.

Next step is to implement small links from the SAMM core model to Agile guidance sections. This allows the core model to stay nice and clean.

Agile guidance coverage



Governance

Strategy & Metrics

- Measure & improve
- Metrics

Education & Guidance

- Training & awareness
- Organization & culture
- Security is a shared responsibility
- Security champions
- Team autonomy

Stream A | Stream B

Design

Threat Assessment

- Threat modeling
- Incremental threat modeling

Security Requirements

- Software requirements
- Selecting and preparing requirements
- Requirements in stories

Stream A | Stream B

Verification

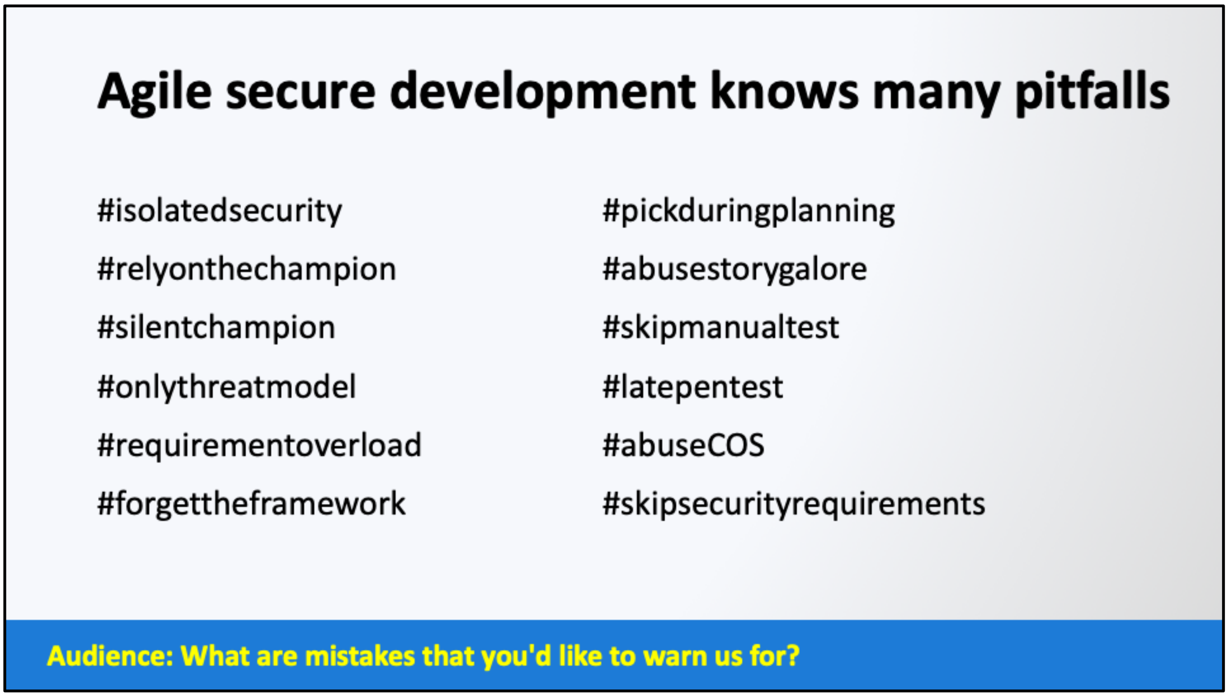
Requirements-driven Testing

- Misuse/abuse testing
- Abuse stories

Security Testing

- Scalable baseline
- Deep understanding
- Agile testing

Stream A | Stream B



Agile secure development knows many pitfalls

#isolatedsecurity

#relyonthechampion

#silentchampion

#onlythreatmodel

#requirementoverload

#forgettheframework

#pickduringplanning

#abusestorygalore

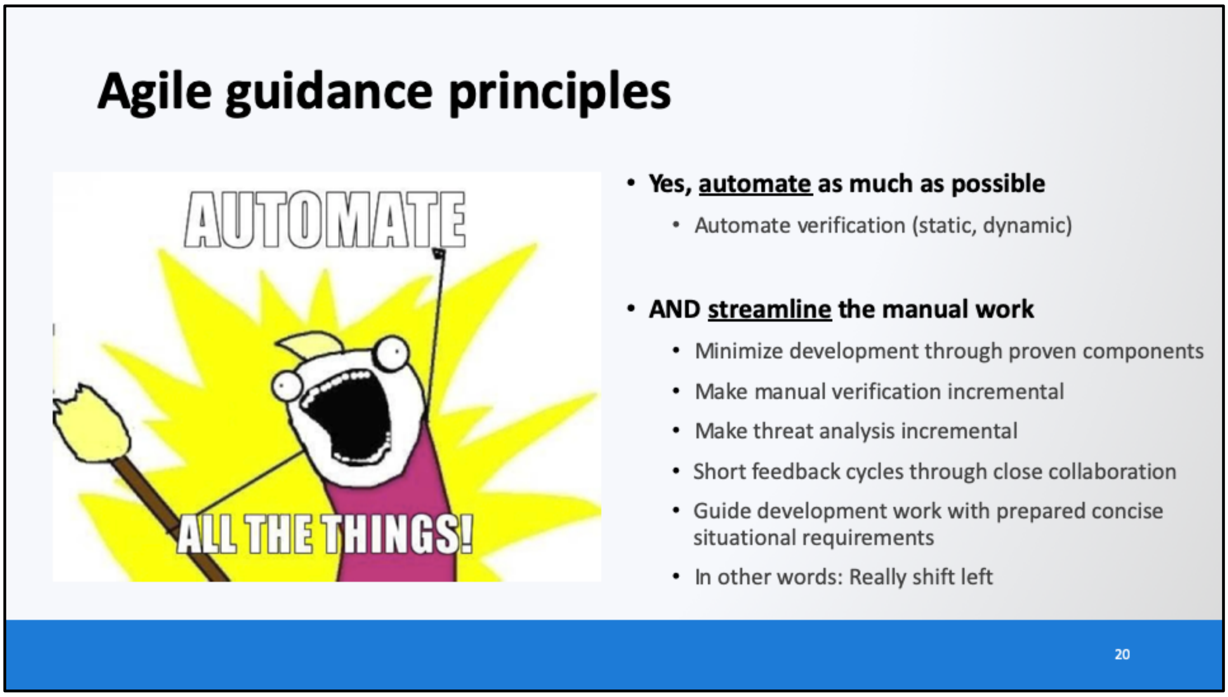
#skipmanualtest

#latepentest

#abuseCOS

#skipsecurityrequirements

Audience: What are mistakes that you'd like to warn us for?



AUTOMATE
ALL THE THINGS!

Agile guidance principles

- **Yes, automate as much as possible**
  - Automate verification (static, dynamic)
- **AND streamline the manual work**
  - Minimize development through proven components
  - Make manual verification incremental
  - Make threat analysis incremental
  - Short feedback cycles through close collaboration
  - Guide development work with prepared concise situational requirements
  - In other words: Really shift left

20

Of course: you need to automate testing as much as possible, both static and dynamic testing. So you need tools to scan your code and tools that try to break into your system. But then you're not done because there's still a lot of manual work that needs to take place.

The trick is to rely on expertise as little as possible by being more efficient, also to prevent people burning out. You sometimes here stories of one security expert for every 50 developers.

One way to become more efficient is to reduce the size of the whole security problem by: Building on proven technology: so you save time but mostly: you have less security to worry about.

Manual verification: code review and penetration testing

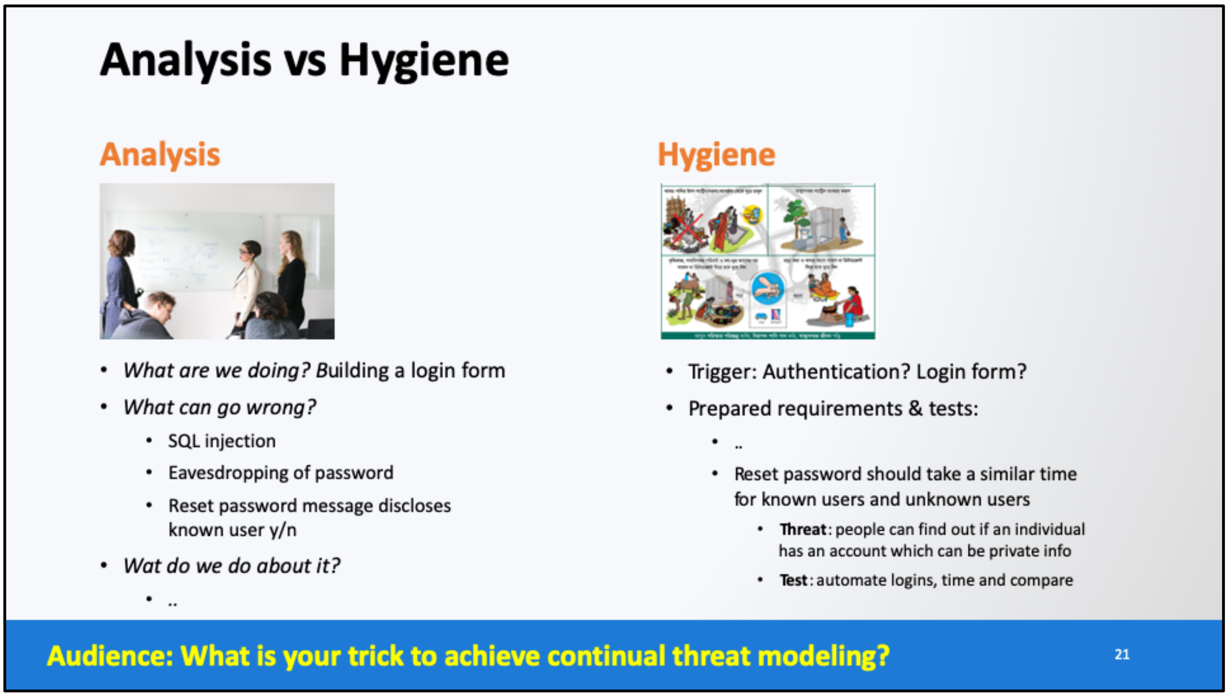
Close collaboration instead of quality gates. You don’t have time to go back and forth between development and test.

Instructions -> so the team doesn’t need constant expert help

The goal is to minimize cognitive load

In other words: there is no time in a sprint to get the development work done and then send it to security for testing. You want to shift left.This puts security professionals much more in an advisory and supportive role than in a role of a quality gate. This also means that all developers and testers should own security, including the product owner.

Let’s look at one of the key things you can do to streamline the manual work, and that has to do with Threat analysis vs hygene.



Analysis vs Hygiene

Analysis

- *What are we doing?* Building a login form
- *What can go wrong?*
  - SQL injection
  - Eavesdropping of password
  - Reset password message discloses known user y/n
- *Wat do we do about it?*
  - ..

Hygiene

- Trigger: Authentication? Login form?
- Prepared requirements & tests:
  - ..
  - Reset password should take a similar time for known users and unknown users
    - **Threat**: people can find out if an individual has an account which can be private info
    - **Test**: automate logins, time and compare

Audience: What is your trick to achieve continual threat modeling?

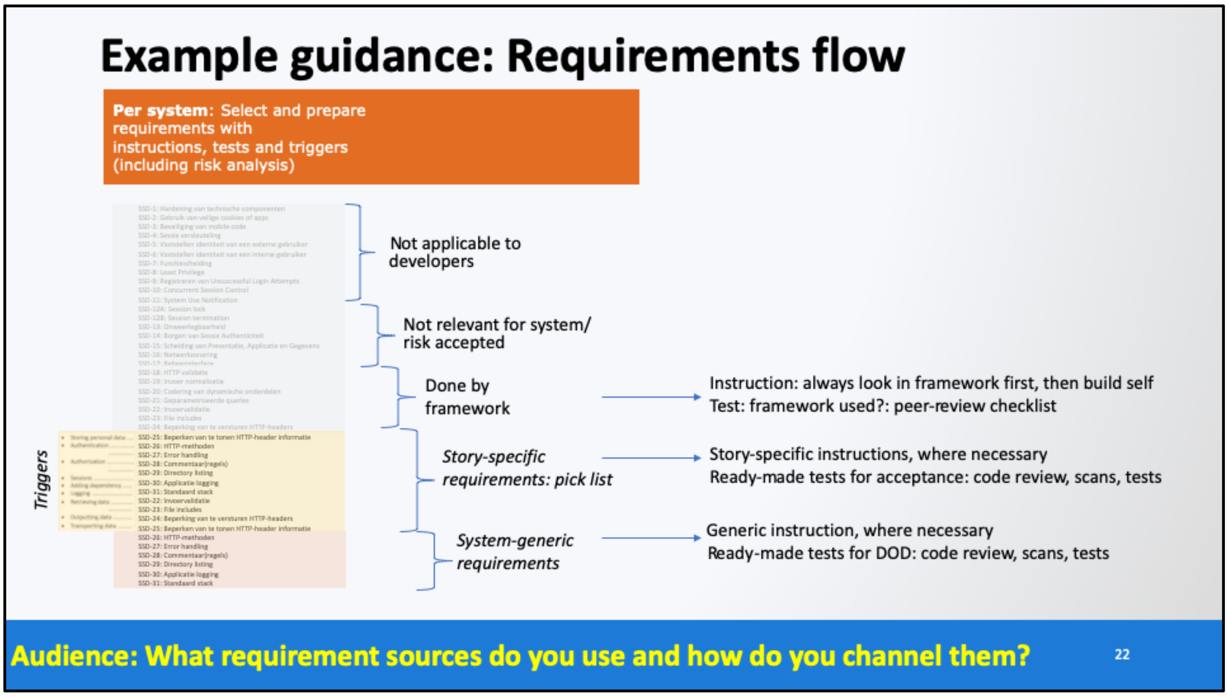
21

This illustrates minimizing the dependency on security expertise, and the dependency of being smart and alert all the time.

If you are having dinner you first wash your hands. Such situational instructions are there to save time in thinking about what you need to do, and they also prevent things to be forgotten.

Please note that threat modelling is very useful. The point here is that if you can make it happen, it is helpful to not be depending on such analysis for trivial things. You shouldn't need to threat model everything because many security countermeasures are always required.

There is a section by the way on incremental threat modeling in the Agile guidance.



Example guidance: Requirements flow

Per system: Select and prepare requirements with instructions, tests and triggers (including risk analysis)

SSD-1: Hardening van technische componenten
SSD-2: Gebruik van veilige cookies of apps
SSD-3: Beveiliging van mobile code
SSD-4: Sessie versleuteling
SSD-5: Vaststellen identiteit van een externe gebruiker
SSD-6: Vaststellen identiteit van een interne gebruiker
SSD-7: Functiescheiding
SSD-8: Least Privilege
SSD-9: Registreren van Unsuccessful Login Attempts
SSD-10: Concurrent Session Control
SSD-11: System Use Notification
SSD-12A: Session lock
SSD-12B: Session termination
SSD-13: Onweerlegbaarheid
SSD-14: Borgen van Sessie Authenticiteit
SSD-15: Scheiding van Presentatie, Applicatie en Gegevens
SSD-16: Netwerkzonering
SSD-17: Beheerinterface
SSD-18: HTTP validatie
SSD-19: Invoer normalisatie
SSD-20: Codering van dynamische onderdelen
SSD-21: Geparametriseerde queries
SSD-22: Invoervalidatie
SSD-23: File includes
SSD-24: Beperking van te versturen HTTP-headers
SSD-25: Beperken van te tonen HTTP-header informatie
SSD-26: HTTP-methoden
SSD-27: Error handling
SSD-28: Commentaar(regels)
SSD-29: Directory listing
SSD-30: Applicatie logging
SSD-31: Standaard stack
SSD-22: Invoervalidatie
SSD-23: File includes
SSD-24: Beperking van te versturen HTTP-headers
SSD-25: Beperken van te tonen HTTP-header informatie
SSD-26: HTTP-methoden
SSD-27: Error handling
SSD-28: Commentaar(regels)
SSD-29: Directory listing
SSD-30: Applicatie logging
SSD-31: Standaard stack

Triggers

- Storing personal data
- Authentication
- Authorisation
- Sessions
- Adding dependency
- Logging
- Retrieving data
- Outputting data
- Transporting data

Not applicable to developers

Not relevant for system/ risk accepted

Done by framework → Instruction: always look in framework first, then build self
Test: framework used?: peer-review checklist

Story-specific requirements: pick list → Story-specific instructions, where necessary
Ready-made tests for acceptance: code review, scans, tests

System-generic requirements → Generic instruction, where necessary
Ready-made tests for DOD: code review, scans, tests

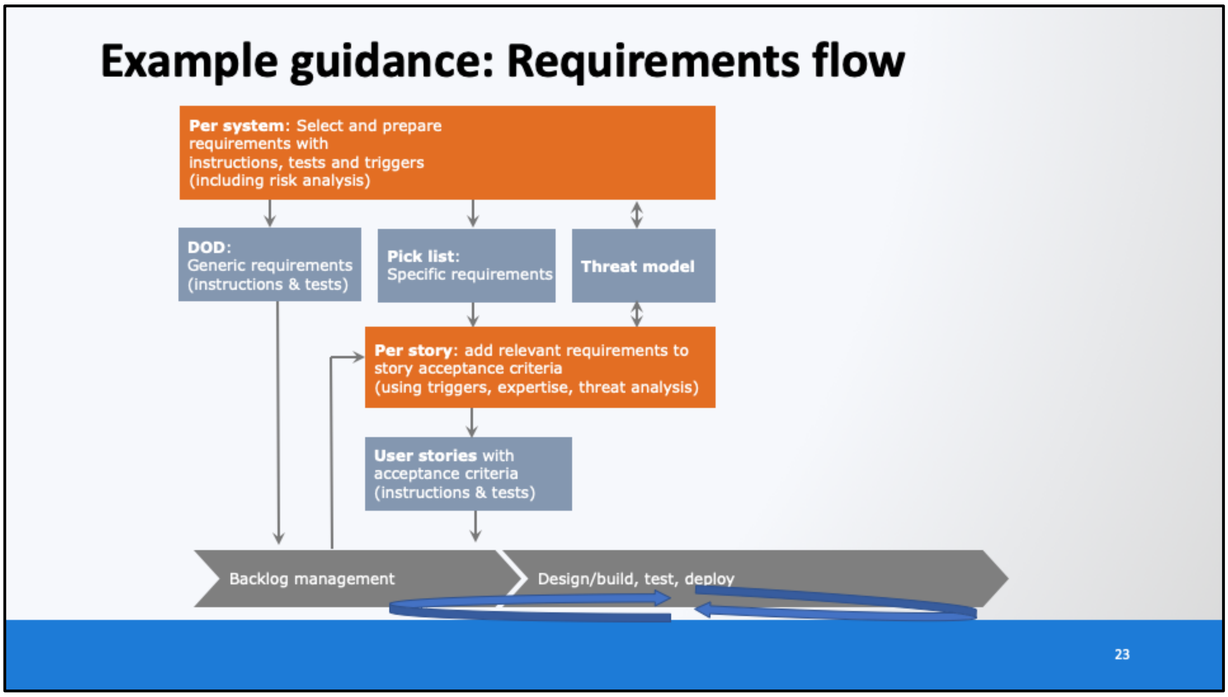
Audience: What requirement sources do you use and how do you channel them?

22

Let’s have a look at an example guidance : how you deal with requirements.

The key is to minimize the daily instructions that developers need to work with, and to let as many as possible depend on the type of task they are working on, for them to be effective.

Audience question: channel = using a wiki, or cards, or tools like SKF or iriuskrusk. Or did you build something yourself. ?
I suspect that Excel is going to be one of the answers.



Example guidance: Requirements flow

Per system: Select and prepare requirements with instructions, tests and triggers (including risk analysis)

DOD: Generic requirements (instructions & tests)

Pick list: Specific requirements

Threat model

Per story: add relevant requirements to story acceptance criteria (using triggers, expertise, threat analysis)

User stories with acceptance criteria (instructions & tests)

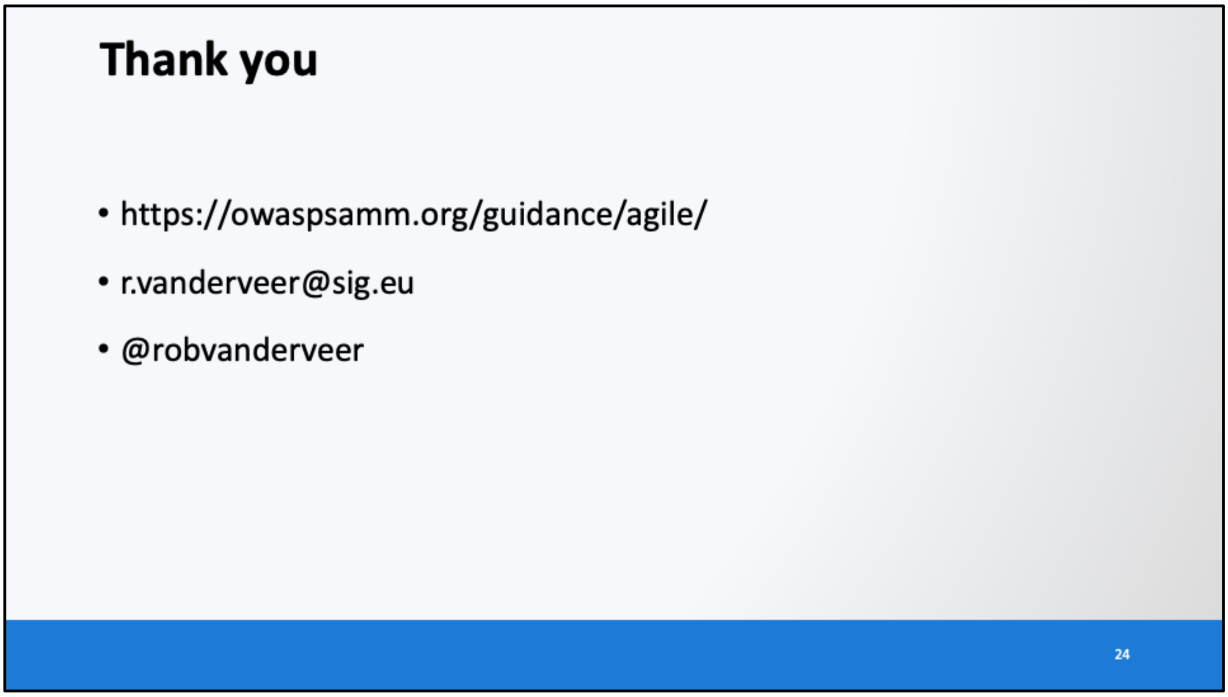
Backlog management

Design/build, test, deploy

23

Tthe treatment of stories happens during creation of the story, and during backlog refinement.
It should not be done during the planning meeting:.

Definition of Ready: security checks should be there,
Definition of Done: security validation needs to be done



Thank you

- https://owaspsamm.org/guidance/agile/
- r.vanderveer@sig.eu
- @robvanderveer

24

With that we are done discussing the Agile guidance. Thank you very much for your joining and if there's time we can take some more questions.

I would welcome your feedback on the notes and your own experiences very much. One final word from me: we talked about security, and this is an application security conference, but everything I have been saying should go for the other quality aspects of software engineering: maintainability, privacy, reliability, and performance efficiency. My personal overarching advice is to see if you can make your security program a software quality program. Good luck. Thank you.