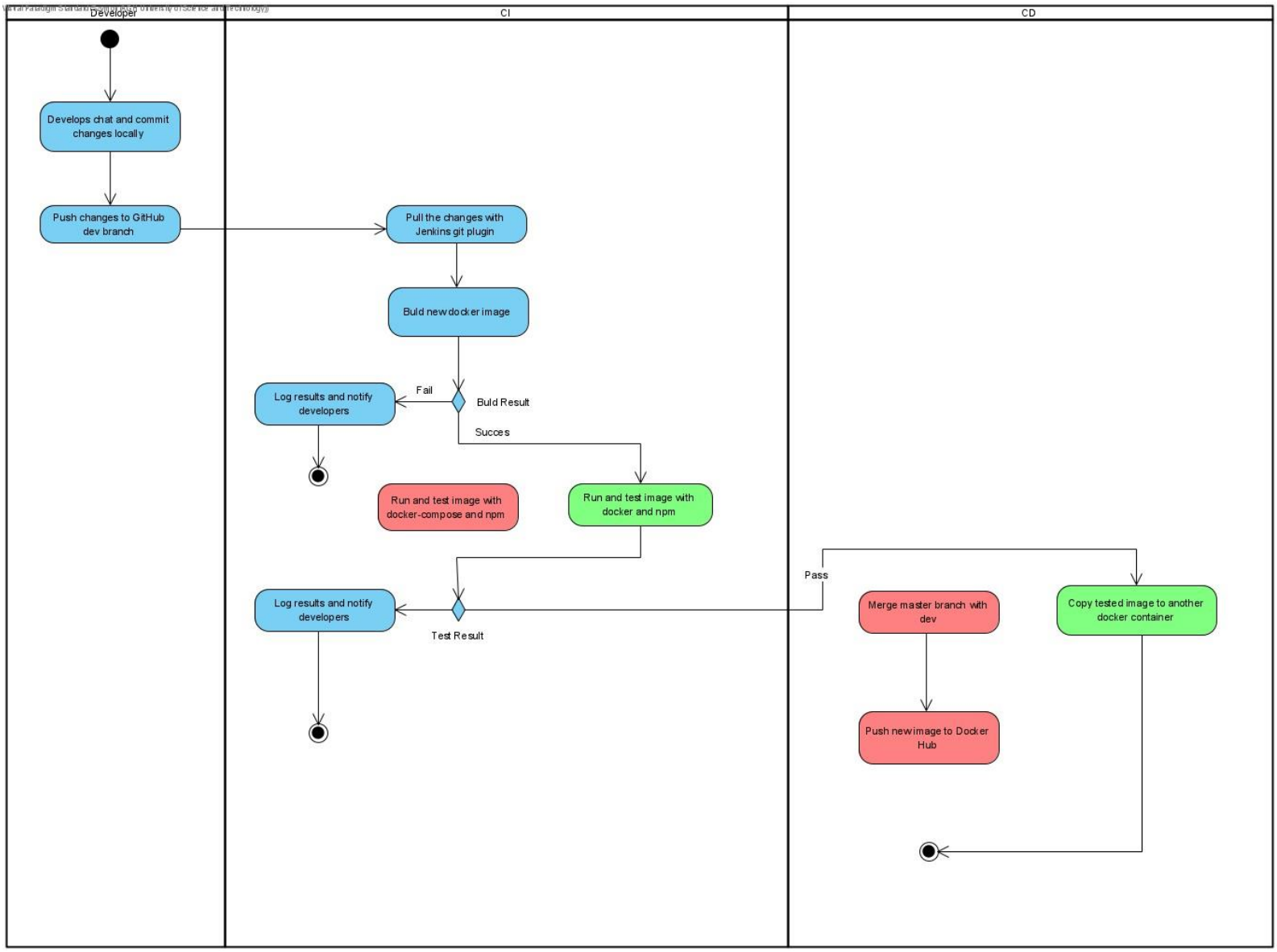
Socket.io-Messenger release pipeline plan v.2

Author: **Szymon Karpęcki**

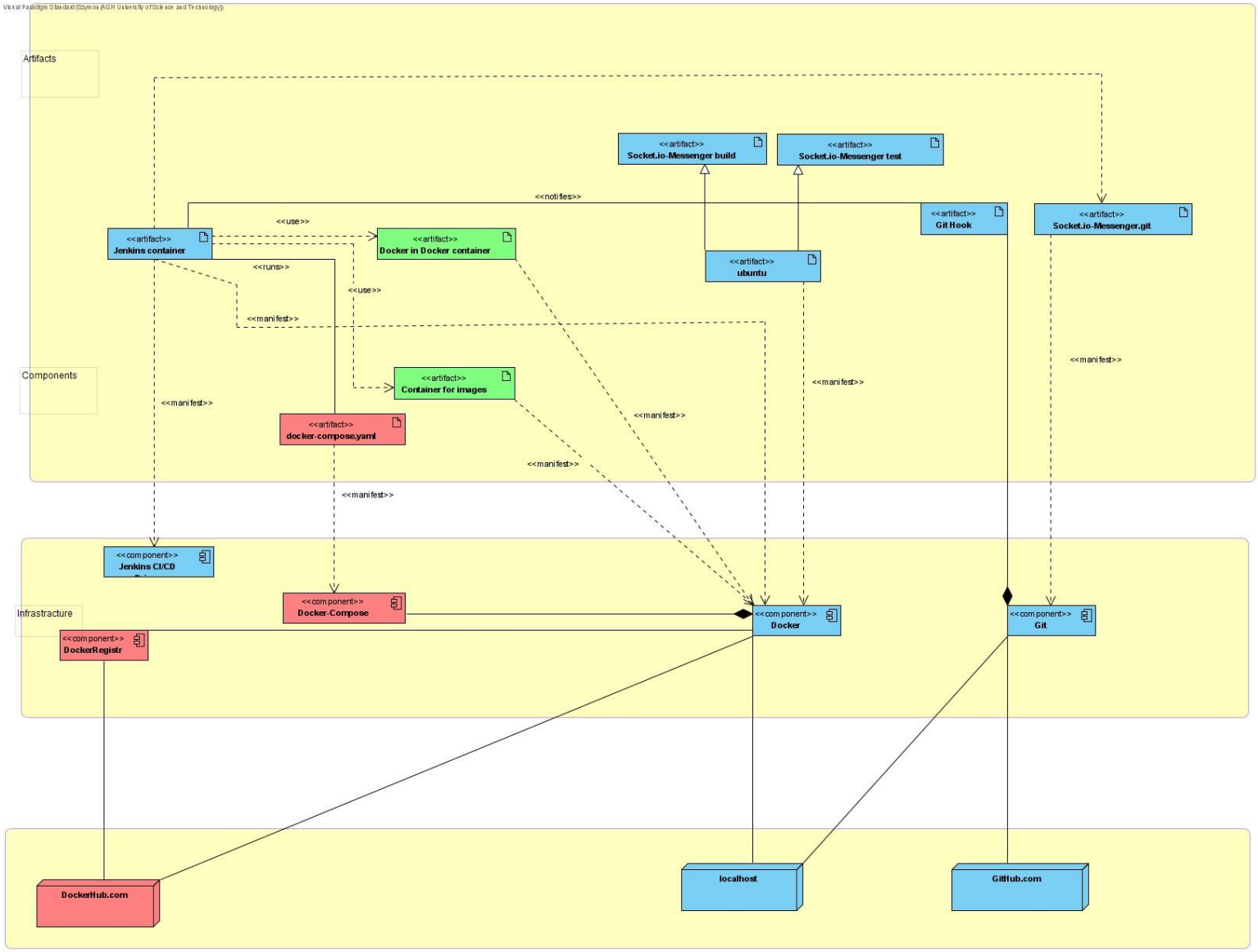
Technologies used:

- Git (GitHub)
- Jenkins
- Docker
- Docker-Composteli
- DockerHub
- Node JS

1. UML Activity Diagram



2. UML Deployment Diagram



3. Porównanie

Activity diagram:

Nazwa kroku	Technologia	Link	Nr. Linii	Komentarz
Push changes to github repo -> Pull the changes with Jenkins git plugin	Git webhook	N/A	N/A	Konfigurowane na maszynie z Jenkinsem
Build new docker image	Docker(file) w DIND i krok w Jenkins(file)	Dockerfile	Całość	Zgodnie z planem
		Jenkinsfile	10-14	
Run and test image with docker and npm	Docker(file) w DIND i krok w Jenkins(file)	Dockerfile	Całość	W planie był docker-compose, tym niemniej prostsze okazało się użycie dockerfile
		Jenkinsfile	16-21	
Publish….	Krok w Jenkins(file)	Jenkinsfile	22-28	Planowo miał być merge do mastera oraz publikacja do dockerhuba. W praktyce było to skopiowanie działającego obrazu do innego kontera dockerowego

Deployment diagram:

Nazwa artefaktu	Technologia	Link	Nr. Linii	Komentarz
Jenkins container	Jenkins+Docker	Jenkinsfile	Calosc	Zgodnie z planem
docker-compose.yaml	Docker compose	-	-	Nieziamplementowane. Wykorzystano dockerfile.
DIND conatiner	Docker	Jenkins docs	pkt. 1-4 (On macOS and Linux)	Nieujęty w pierwotnym planie, mimo iż konieczny.
Container for images	Docker	Jenkinsfile	24	Nieujęty w pierwotnym planie, zastępuje Dockerhuba
Git hook	Bash	Github+Jenkins +Webhookrelay	-	Zgodnie z planem
Ubuntu	Docker	Dockerfile	1	Zgodnie z planem
Chat Build	Docker	Dockerfile	Całość	Zgodnie z planem
Chat test	Docker	Dockerfile	Całość	Zgodnie z planem