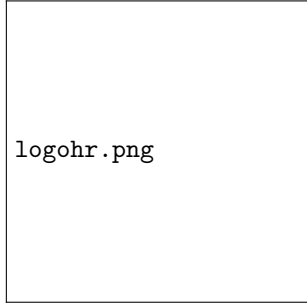


Verslag Tinlab Advanced Algorithms

Galvin Bartes 0799967
176-671

24 oktober 2023



logohr.png

Inhoudsopgave

1	Inleiding	2
2	Theoretisch kader	3
2.1	Begrippen, tools en literatuur	4
2.2	Rampen	7
2.2.1	Therac-25	8
2.2.2	Ethiopian Airlines Flight 302,boeing 737 crashes	9
2.2.3	China explosie 2015 Tianjin	9
2.2.4	tesla autopilot crashes	10
2.2.5	de malimissie	11
2.2.6	militair overleden door schietoefening in ossendrecht	12
2.2.7	schipholbrand	12
2.2.8	explosie in beirut	13
2.2.9	Bijlmerramp	13
2.2.10	1951	13
2.2.11	slmramp	14
2.2.12	Tsjernobyl	14
2.2.13	oekraïne powergrid	14
2.2.14	Stint	15
2.2.15	Safety critical systems	16
3	Logica	18
3.1	Propositie logica	18
3.2	Predicaten logica	18
4	Computation tree logic	19
4.1	Algemeen	19
4.2	De Kripke structuur	19
4.3	Soorten transitie	20
4.4	Tijd	20
4.5	Guards en invarianten	21
4.6	Safety	21
4.7	Fairness	23
4.8	Deadlock	24
4.9	Zeno gedrag	24
5	Verificatie	24
6	Conclusie	26
A	Edmund Clarke model checking samenvatting	27

B	Ontwikkelgeschiedenis	30
B.1	Ontwerpen: 2 juni	34
B.2	Ontwerpen: 2 mei	37
B.3	Ontwerpen:4 juli	41
B.4	Ontwerpen: 6 juni	48
B.5	Ontwerpen: 8 juni	52
B.6	Ontwerpen: 10 april	56
B.7	Ontwerpen: 12 juni	59
B.8	Ontwerpen: 12 september	65
B.9	Ontwerpen: 13 april	66
B.10	Ontwerpen: 13 juni	69
B.11	Ontwerpen: 15 april	74
B.12	Ontwerpen:16 juli	77
B.13	Ontwerpen: 17 april	81
B.14	Ontwerpen: 1 mei	84
B.15	Ontwerpen: 19 september versie 2	89
B.16	Ontwerpen: 20 september	92
B.17	Ontwerpen: 22 mei	95
B.18	Ontwerpen: 23 juni	99
B.19	Ontwerpen: 26 april	105
B.20	Ontwerpen: 30 april versie 1	108

1 Inleiding

Algemeen Het ministerie van verkeer en Waterstaat wil in het kader van het klimaatakkoord en onderzoek laten uitvoeren naar de staat van het sluisenpark in Nederland. Het onderzoek moet zich richten op het ontwerpen en ontwikkelen van een geautomatiseerd sluismodel dat geschikt is voor een brede toepassing. In het onderzoek moet naar voren komen wat de huidige staat is van de sluisen met oog op veiligheid, efficiëntie, capaciteit, onderhoud, duurzaamheid en automatisering. Het onderzoek geeft aan hoe een volledig model worden opgeleverd opdat ontwerp van verschillend volledig geautomatiseerde sluisen in de toekomst geautomatiseerd kunnen worden.

Probleemanalyse Na grondige analyse van het Nederlandse sluisenpark is gebleken dat renovatie van een groot aantal sluisen noodzakelijk is. Uit een eerste verkenning is gebleken dat het gecombineerd renoveren en automatiseren van het Nederlandsesluisenpark een aanzienlijke verbetering kan opleveren t.a.v. Op het ministerie van infrastructuur en waterstaat is helaas onvoldoende kennis van ict en systemen aanwezig om een ander uit te voeren

Waarom nu In het kader van het onlangs afgesloten klimaatakkoord heeft de Nederlandse overheid daarom besloten over te gaan tot een ingrijpende renovatie van diverse sluisen die ons land rijk is.

Gewenst resultaat Wij vragen u een model (of een onderling samenhangend aantal modellen) aan te leveren, opdat ontwerpen van verschillende, volledig geautomatiseerde sluisen in de toekomst gerealiseerd kunnen worden. Zoals gesteld in de brief is het de bedoeling dat een sluis gemodelleerd wordt dat bewezen kan worden dat de te bouwen sluis een aantal eigenschappen bezit.

Ons doel is een uppaal model van een sluis op te leveren. We willen een fysiek systeem vastleggen in software, ofwel een domein uit de echte wereld overplaatsen naar het conditionele. De fenomenen uit de echte wereld worden gemonitord met sensoren. De fenomenen uit de wereld worden kenbaar gemaakt aan het softwaresysteem in de vorm van variabele data. Welke data wordt opgevangen, opgeslagen en uitgelezen wordt vastgelegd in de requirements. De manier waarop dit gebeurt wordt vastgelegd in specificaties. De requirements worden verkregen door requirements engineers. Dit varieert van concepten en best-practices uit observaties, interviews, stakeholders analysis, focus group, document analysis, het verkennen van user requirements, task analysis, surveys en problem analysis. Requirements worden onderverdeeld in functioneel en niet-functioneel. Functionele requirements omschrijven de klantwens, ofwel functie en gedrag. Niet-functionele requirements/eisen zijn beperkt tot vereisten die aan systemen worden opgelegd. Ze hebben betrekking op kwaliteitsattributen als: schaalbaarheid, onderhoudbaarheid, beveiliging, betrouwbaarheid. Belangrijk is de vraag wat is een goed model. Voor het testen van een goed model of een specificatie zijn verschillende technieken. In de biomedische wereld wordt er een onderscheid gemaakt tussen in vivo "levendige in vitro afgeschieden experimenten in silico een gecomputeriseerd model".

Scope Het gaat om het simuleren van een geautomatiseerde sluis. Wat voor type sluis wordt niet gemeld en ook niet uit welke onderdelen. Belangrijk is dat het model werkt en dat het voldoet aan de eisen die gebaseerd zijn op basis van literatuuronderzoek, observatie, interviews, brainstorming of een andere vorm van requirements elicitation.

Onderzoeksvragen Hoe kan een geautomatiseerde sluis worden gemodeleerd met oog op ontwikkelen onderhoudskosten, veiligheid, efficiëntie en capaciteit

1. Welke requirements en kwaliteitseisen komen naar voren bij de analyse van een rampenonderzoek
2. Welke veiligheidseisen er zijn voor sluizen in Nederland.
3. Hoe kan in uppaal een model worden getest dat voldoet aan de requirements/eisen volgens het rampenonderzoek?

Design goals Het systeem moet minimaal aan de volgende prestatie eisen voldoen

1. Requirements gebaseerd op rampenanalyse
2. Model testbaar in uppaal

Methodologie [?]

Afbakening

Leeswijzer In de methodologie wordt de lezer uitgelegd met welke methoden de onderzoeksvragen zijn beantwoord. In het hoofdstuk Onderzoek worden alle resultaten behandeld die naar voren zijn gekomen bij het deskresearch. De analyse van de verzamelde data wordt gedaan in het hoofdstuk analyse. Hierin wordt behandeld zoekopdracht naar IoT cloud platforms, feature extractie, prijs-berekening en prijs-feature vergelijking. In het ontwerp komen de uml diagrammen en systeemschetsen naar voren. In de hoofdstukken Prototype, IoT cloud en Firmware wordt de implementatie behandeld van het IoT cloud platform in een bestaand project.

2 Theoretisch kader

In het eerste hoofdstuk is duidelijk geworden wat de onderzoeksvraag is, namelijk 'Hoe kan een geautomatiseerde sluis worden gemodeleerd met oog op ontwikkelen- en onderhoudskosten, veiligheid, efficiëntie en capaciteit'. Door de toenemende complexiteit van systemen is het gebruik van modellen en de toepassing van timebased model checking op industriële controle systemen een manier van modelleren van het systeem en de requirements zodat er een bijdrage kan worden geleverd aan de acceptatie van simulatie-/modeltechniek voor de industrie.[?]. Of dit ook het geval is bij het modelleren van sluizen is nu de vraag.

De bestudering van rampen aan de hand van het vier-variabelen model biedt maakt het analyseren mogelijk van rampsituaties. Van een aantal rampen is een beschrijving gegeven met datum, plaats en oorzaak. De analyse van de 4-variabelen modellen zal gebruikt worden voor de requirementsdefinitie, ontwerp en ontwikkeling van het sluismodel.

De verschillende factoren en achtergronden die samenhangen met het modelleren van een sluis zullen in dit hoofdstuk toegelicht worden. Bovendien worden er hypothesen gevormd die de basis vormen voor de beantwoording van de onderzoeksvraag.

2.1 Begrippen, tools en literatuur

Wat is uppaal Uppaal is een geïntegreerde toolomgeving voor het modelleren, simuleren en verifiëren van real-time systemen, gezamenlijk ontwikkeld door Basic Research in Computer Science aan de Universiteit van Aalborg in Denemarken en de afdeling Informatietechnologie aan de Universiteit van Uppsala in Zweden. Het is geschikt voor systemen die kunnen worden gemodelleerd als een verzameling niet-deterministische processen met een eindige controlestructuur en klokken met reële waarde, die communiceren via kanalen of gedeelde variabelen. Typische toepassingsgebieden zijn met name real-time controllers en communicatieprotocollen, waarbij timingaspecten van cruciaal belang zijn.

Wat is statistical model checking? Dit verwijst naar verschillende technieken die worden gebruikt voor de monitoring van een systeem. Daarbij wordt vooral gelet op een specifieke eigenschap. Met de resultaten van de statistieken wordt de juistheid van een ontwerp beoordeeld. Statistisch model checking wordt onder andere toegepast in systeembioogie, software engineering en industriële toepassingen. [?]
[?]

Waarom gebruiken we statistisch model checking? Om de bovenstaande problemen te overwinnen stellen we voor om te werken met Statistical Model Checking, een aanpak die onlangs is voorgesteld als alternatief om een uitputtende verkenning van de toestandsruimte van het model te vermijden. Het kernidee van de aanpak is om een aantal simulaties van het systeem uit te voeren, deze te monitoren en vervolgens de resultaten uit het statistische gebied te gebruiken (inclusief het testen van sequentiële hypothesen of Monte Carlo-simulaties) om te beslissen of het systeem aan de eigenschap voldoet of niet. mate van vertrouwen. Van nature is SMC een compromis tussen testen en klassieke modelcontroletechnieken. Het is bekend dat op simulatie gebaseerde methoden veel minder geheugen- en tijdsintensief zijn dan uitputtende methoden, en vaak de enige optie zijn. [?] Alternatieve tools voor Uppaal zijn Asynchronous Events, Vesta en MRMC.

MODE CONFUSION Mode confusion treed op als geobserveerd gedrag van een technisch systeem niet past in het gedragspatroon dat de gebruiker in zijn beeldvorming heeft en ook niet met voorstellingsvermogen kan bevatten.

Wat is automatiseringsparadox Gemak dient de mens. Als er veel energie wordt gestoken in de ontwikkeling van hulpmiddelen die taken van werknemers overnemen heeft dat tot resultaat dat veel productieprocessen worden geautomatiseerd. De vraag is dan of vanuit mechnisch wereldpunt de robot niet de rol van de mens overneemt en of de mens nog de kwaliteiten heeft om het werk zelf te doen. [?]
[?] [?]

4 variabelen model Er zijn veel veiligheidskritische computersystemen nodig voor het bewaken en besturen van fysieke processen. Het viervariabelenmodel, dat al bijna met succes in de industrie wordt gebruikt veertig jaar, helpt het gedrag van en de grenzen tussen het fysieke te verduidelijken processen, invoer-/uitvoerapparaten en software. [?]

Het 4 variabelen model kort toegelicht Monitored variabelen: door sensoren gekwantificeerde fenomenen uit de omgeving, bijv temperatuur

Controlled variabelen: door actuatoren bestuurd fenomenen uit de omgeving Gecontroleerde variabelen kunnen bijvoorbeeld de druk en de temperatuur zijn in een kernreactor, terwijl gecontroleerde

variabelen ook visuele en hoorbare alarmen kunnen zijn als het uitschakelsignaal dat een reactorsluiting initieert; wanneer de temperatuur of druk bereikt abnormale waarden, de alarmen gaan af en de uitschakelprocedure wordt gestart

Input variabelen: data die de software als input gebruikt Hier modelleert IN de ingangshardware-interface (sensoren en analoog-naar-digitaal-omzetters) en relateert waarden van bewaakte variabelen aan waarden van invoervariabelen in de software. De invoervariabelen modelleren de informatie over de omgeving die beschikbaar is voor de software. Bijvoorbeeld, IN zou een druksensor kunnen modelleren die temperatuurwaarden omzet in analoge spanningen; Deze spanningen worden vervolgens via een A/D-omzetter omgezet in gehele waarden die zijn opgeslagen in een register dat toegankelijk is voor de computer software.

Output variabelen: data die de software levert als output De uitgangshardware-interface (digitaal-naar-analoog converters en actuatoren) is gemodelleerd door OUT, dat waarden van de uitvoervariabelen van de software relateert aan waarden van gecontroleerde variabelen. Een uitvoervariabele kan bijvoorbeeld een Booleaanse variabele zijn die door de software is ingesteld met de begrippen dat de waarde true aangeeft dat een reactorsluiting zou moeten plaatsvinden en de waarde false geeft het tegenovergestelde aan

6 Variable model Een uitbreiding van een 4-variabelen model is het 6-variabelen model. Optitatieve statements omschrijven de omgeving zoals we het willen zien vanwege de machine. Indicatieve statements omschrijven de omgeving zoals deze is los van de machine. Een requirement is een optitatief statement omdat ten doel heeft om de klantwens uit te drukken in een softwareontwikkel project. Domein kennis bestaat uit indicatieve uitspraken die vanuit het oogpunt van software ontwikkeling relevant zijn. Een specificatie is een optitatief statement met als doel direct implementeerbaar te zijn en ter verondersteuning van het nastreven en realiseren van de requirements. Drie verschillende type domeinkennis: domein eigenschappen, domein hypothesen, en verwachtingen. Domein eigenschappen zijn beschrijvende statementsover een omgeving en zijn feiten. Domein hypothesen zijn ook beschrijvende uitspraken over een omgeving, maar zijn aannames. Verwachtingen zijn ook aannames, maar dat zijn voorschrijvende uitspraken die behaald worden door actoren als personen, sensoren en actuators.

World and machine samenvatting Inderdeel van deze cursus is de studie van het artikel World as a machine [?] Software engineering maakt het gebruik van fysieke apparaten mogelijk. Het is een beschrijving van de machine. Het doel van een machine ligt in de wereld. Commando afhandeling wordt niet beoordeeld door examinering van de software, maar door te kijken naar de kwaliteit van de geprogrammeerde documenten en het gebruik, gemak en voldoening voor de operators. Het probleem is de requirement dat ligt in de wereld en de machine is de oplossing die we bedenken. De relatie tussen machine en wereld blijkt uit de volgende 4 facetten:

- 1 modelling facet:
- interface facet:
- engineering facet:
- problem facet: requirements, specificaties en programmas zijn subsets van domeinen bestaande uit wereldlijke en machinale fenomenen. Decompositie van probleem verloopt parrallel en niet hiërarchisch.

denial bij knowledge: net meer het wiel uitvinden denial by hacking: obsessief toegewijs aan interactie met computers denial by abstraction: softwareproblemen kunnen bevangen worden in enkele simpele woorden. Het moeilijke is problemen op te splitsen. Denial by vagueness: vaag taalgebruik om probleem te verhullen

Formal recognizable description Gebruikte event als grondterm en geen english noun. Dit is recognizable. De wereld is niet strongly typed verschil tussen aannemen(optatief) en willen zien(indicatief)

Systeem vs software requirement Ter verduidelijking van het verschil tussen systeemrequirement en softwarerequirement hieronder een overzicht:

Een system requirement is een uitspraak over wereld fenomenen (gedeeld of niet) of doelen die bereikt moeten worden. Een systeem requirement is met enige regelmaat informeel, niet precies geformuleerd. Systemen gaan een zekere interactie aan met hun omgeving: Sensoren: meten fenomenen uit de omgeving (temperatuur, druk, licht, geluid, etc.). Actuatoren: veranderen iets in de omgeving (mechanische, elektrisch, pneumatisch, etc.)

Een software requirement/specificatie is een uitspraak over gedeelde fenomenen of doelen die de machine moet bereiken middels de onderdelen waar die machine uit of middels de fenomenen waar de machine controle over heeft. Is doorgaans preciezer, meetbaar, exact geformuleerd.

Software kan niet direct communiceren met de buitenwereld. Snapt derhalve niets van de buitenwereld. Kan alleen maar bestaan in en communiceren met het systeem.

Requirementsengineering Om de juiste requirements te verzamelen en selecteren hebben we meer kennis nodig van de methoden hiervoor gebruikt in het domein van requirementsengineering. [?] [?] [?]
[?]

[?] [?].

what is a good software specification

- Een goed model heeft een duidelijk gespecificeerd modelleringsobject, dat wil zeggen dat het duidelijk is wat het model beschrijft.
- Een goed model heeft een duidelijk omschreven doel en draagt (idealerweise) bij aan de realisatie van dat doel.
- Een goed model is traceerbaar: elk structureel element van een model (1) komt overeen met een aspect van het modelobject, of (2) codeert voor impliciete domeinkennis, of (3) codeert voor een aanvullende aanname.
- Een goed model is waarheidsgetrouw: relevante eigenschappen van het model moeten ook worden overgedragen op (behouden voor) het object van modellering.
- Een goed model is eenvoudig (maar niet te eenvoudig).
- Een goed model is uitbreidbaar en herbruikbaar, dat wil zeggen dat het is ontworpen om te evolueren en te worden gebruikt buiten het oorspronkelijke doel.
- Er is een goed model ontworpen en gecodeerd voor interoperabiliteit en het delen van semantiek.

[?]. Meer artikelen zijn: [?] [?] [?] [?] [?] [?] [?]

Wat is een sluis Een sluis is ook een scheiding tussen 2 wateren, maar met deuren. Hierdoor is het mogelijk het waterpeil te beïnvloeden. Sluizen reguleren het waterpeil zodat schepen kunnen passeren. Een stuw is een vaste of beweegbare afdamming tussen 2 wateren. [?]

[?] [?] [?] Een model van een sluispassage kan worden gemodelleerd met procestates zoals het model aangeboden door Rijkswaterstaat



[?, p.79 –113] [?, p.159],

Recente ontwikkelingen op het gebied van sluisautomatisering Het ministerie van verkeer en Waterstaat wil in het kader van het klimaatakkoord en onderzoek laten uitvoeren naar de staat van het sluizenpark in Nederland. Het onderzoek moet zich richten op het ontwerpen en ontwikkelen van een geautomatiseerd sluismodel dat geschikt is voor een brede toepassing. In het onderzoek moet naar voren komen wat de huidige staat is van de sluizen met oog op veiligheid, efficiëntie, capaciteit, onderhoud, duurzaamheid en automatisering. Het onderzoek geeft aan hoe een volledig model worden opgeleverd opdat ontwerp van verschillend volledig geautomatiseerde sluizen in de toekomst geautomatiseerd kunnen worden.

2.2 Rampen

Voor deze studie is onderzoek gedaan naar verschillende rampen aan de hand van het vier variabelen model. Elke ramp op deze manier categoriseren kan ons helpen te bepalen in hoeverre requirements een rol kunnen spelen in de veiligheid van ons model.

2.2.1 Therac-25

De therac-25. In de periode van Juni 1985 and Januarie 1987 zijn er meerdere ongelukken met dodelijke afloop door de implementatie van de Therac-25 bij de behandelig van huidkanker. De therac-25 is een Medische lineaire versneller. Deze versnellen elektronen om stralen met hoge energie te creëren die tumoren kunnen vernietigen met minimale impact op het omliggende gezonde weefsel. Onderzoekers constateren dat er fouten zijn gemaakt tijdens de (her-)implementatie van systemen uit eerdere productiemodellen. Terwijl de therac 20 afhankelijk was van mechanische vergrendelingen werd er bij de therac-25 software gebruikt.

Enkele daarvan zijn Yakima Valley Memorial Hospital in 1985, East Texas Cancer Center in maart 1986, het East Texas Cancer Center in april 1986 en Yakima Valley Memorial Hospital. Veel fouten in safety-critical systeem. In geval van therac spreken we van een systeemongeluk, complexe interacties tussen verschillende componenten en activiteiten. In het artikel worden enkele ongevallen omschreven. In het tweede geval is er sprake van onvolmaakte microswitches welke een ambigu bericht produceert voor de computer. In het derde geval zijn er open slots in de blocking trays. In het Vierde geval heeft de operator verkeerde prescriptie-data ingevoerd. De operator drukt op return en bevestigt alsnog de invoer. Op een gegeven moment komt er een foutmelding "malfunction 54". De operator was bekend met de machine en drukte op de knop "p" van proceed. In het vijfde geval was er een verkeerde invoer voorschrift data waardoor verkeerde toets werd gedrukt. Na aanpassen werd de return-toets ingedrukt. Er trad een fout op met de melding "malfunction 54". Na reproductie van de melding bleek dat de ionische kamer niet gezouten bleek te zijn. In het zesde geval vergat de operator de files te verwijderen onder de patent. Er werd straling gemeten maar de console gaf aan dat er geen dosisratio was gemeten. De operator drukte op de knop "p" om het proces te pauzeren.

Onderzoekers komen daarom tot de volgende conclusies Software problemen zijn onder andere:

- slechte software engineering/designing praktijken
- er is een machine gebouwd dat afhankelijk is van software voor veiligheidsoperaties
- de fout in de code is niet zo belangrijk als een geheel onveilig ontwerp
- het reinigen van de buigmagneetvariabele in plaats van aan het uiteinde van het frame
- raceconditionering om aan te geven dat het invoeren van het recept nog steeds aan de gang is
- reactie van de gebruiker
- slechte subroutines voor schermvernieuwing die rommel en foutieve informatie op de werkende console achterlieten
- Problemen met het laden van tapes bij het opstarten, waarbij het gebruik van photom-tabellen werd uitgesloten om het interlock-systeem te activeren in het geval van een laadfout in plaats van een checksum

[illegible]

- Onverenigbaar grondgebruik in de nabije omgeving. Veel woonwijken met naar schatting 6000000 bewoners en 500 lokale bedrijven in de buurt van de opslag gevaarlijke stoffen.

2.2.4 tesla autopilot crashes

De veiligheidsrisicos van de tesla lopen dus uiteen. Zo zijn er risico's in de machinelearning technologie waarbij drie stickers op het wegdek de autopilot van een tesla den besluiten van baan te wisselen. [?], [?]. [?]. Een studie door de consumptenbond in de VS toont achteraf aan dat het autopilot systeem van de testla niet failsafe is. Zo zijn de sensoren, gebruikt voor detectie van een bestuurder negatief te beïnvloeden. [?]. Ook zijn er issues zonder een dodelijke afloop maar wel met een grote impact op veiligheid. Zoals problemen met de bluetooth [?], touch screen [?][?], Web-based attack crashes Tesla driver interface. [?]. Zelfs de tesla batterij is veiligheidsvraagstuk geworden [?][?]. Maar ook was een onderzoeker was in staat om persoonlijke details van afgedankte voertuigonderdelen te verkrijgen nadat deze waren afgekeurd vanwege upgrades en reparaties op consumentenvoertuigen. [?] Dan blijkt Data-opslag in de cloud niet altijd bereikbaar. [?] En ook maakt diestall mogelijk softwarefout [?] Of is

aanwezigheid van grote hoeveelheid brandbare materialen, fungeerde als brandstof waardoor de brand zich verder kon ontwikkelen zowel binnen als buiten de cel. Doordat de dakluiken niet werken werd de rook en warmte niet afgevoerd, dit heeft de reddingsoperatie van de bewaarders belemmerd en het mede onmogelijk gemaakt alle celdeuren te openen. Door de schilconstructie van het cellencomplex was het mogelijk dat de brand zich kon uitbreiden naar de andere cellen en naar de gang.

De bewoners van vleugel k zijn in eerste instantie geëvacueerd naar vleugel j in plaats van kruislings naar vleugel A. Door de evacuatie naar vleugel j bleven de bewoners geconfronteerd met de brand, waardoor er onnodig sprake was van angstgevoelens. Daarnaast werden de bewoners onvoldoende geïnformeerd over de actuele situatie tijdens de brand en over de overplaatsing naar andere centra.

Er was geen evacuatieplan dat voorziet in de overplaatsing van gedetineerden naar andere penitentiaire inrichtingen. Bij de evacuatie naar andere detentiecentra is te weinig aandacht geweest voor ademhalingsproblemen van gedetineerden. Ook ging de grote instroom van gedetineerde in andere centra ten koste van kwaliteit van de opvang en nazorg. Zorgverlening te veel afhankelijk van ad-hoc maatregelen. De dienst justitiele inrichtingen en de locatiedirecteur hadden geen duidelijk zicht op welke celbewoner naar welk detentiecentrum was overgeplaatst.

2.2.8 explosie in beirut

Op 23 september 2013 voer een vrachtschip de Rhosus onder Moldavische vlag van Batoemi in Georgië naar Beira in Mozambique met 2.750 ton ammoniumnitraat. Ondanks het ernstige gevaar van het bewaren van deze goederen in de hangar onder ongeschikte klimatologische omstandigheden werden er geen maatregelen genomen. De explosie werd vergeleken met een nucleaire bom. De opslag van explosieve materialen is ontstaan door. Voorafgaand aan de explosie was er een brand in een opslagplaats. Uit mediaberichten bleek dat er een verzoek was ingediend bij de marine instantie om deze goederen onmiddellijk weer te exporteren om de veiligheid van de haven en de mensen die er werken te verzekeren, of om akkoord te gaan om ze te verkopen.

2.2.9 Bijlmerramp

De bijlmerramp [?], vond plaats op 04/10/1994. Op de avond van de 4e oktober 1992 waren er bij het toetel van vliegmaatschappij El Al fluctuaties in de snelheidsregulering, fluctuaties in de voltage electriciteit van motor 3. Doordat de rechtervleugel beschadigd was, was het moeilijker om het vliegtuig recht te houden. Alleen de hoge snelheid zorgde ervoor dat er nog voldoende draagvermogen was. Toen bij het inzetten van de landing de snelheid verlaagd werd, werd het draagvermogen van de rechtervleugel echter dusdanig gering dat het toestel niet meer onder controle te houden was en een duikvlucht naar rechts maakte. [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?]

2.2.10 1951

Op woensdag 25 februari 2009 voltrok zich de ramp met een toestel van Turkish Airlines tijdens vlucht 1951. De automatische reactie van het toestel werd getriggerd door een fout gevoelige radio altimeter waardoor de automatische gashendel de energiemotor op actief stelde. Inadequaat handelen van de piloten ondanks een defecte hoogtemeter en onvolledige instructies van de luchtverkeersleiding [?] [?] [?] [?] [?] [?] [?].

2.2.11 slmramp

Toen het toestel op 07/06/1989 de Anthony Nesty Zanderij naderde, was het daar, anders dan het weerbericht had voorspeld, mistig. Het zicht was evenwel niet zo slecht dat er niet op zicht kon worden geland. Gezagvoerder Will Rogers besloot echter via het Instrument Landing System (ILS) te landen, hoewel dit niet betrouwbaar was en hij voor zo'n landing ook geen toestemming had. De gezagvoerder brak drie landingspogingen af. Bij de vierde poging negeerde de bemanning de automatische waarschuwing (GPWS) dat het toestel te laag vloog. Het toestel raakte op 25 meter hoogte twee bomen. Het rolde om de lengteas en stortte om 04.27 uur plaatselijke tijd ondersteboven neer. Uit onderzoek bleek dat de papieren van de bemanning niet in orde waren door nalatigheid in de crew-member screening. Geconcludeerd werd dat de gezagvoerder roekeloos had gehandeld door voor een ILS-landing te kiezen terwijl hij daar geen toestemming voor had, en door onvoldoende op de vlieghoogte te hebben gelet. De SLM werd verweten de kwalificaties van de bemanning onvoldoende te hebben gecontroleerd. De Oorzaak van de ramp bleek achteraf het roekeloos besturen door de kapitein onder de minimum hoogte leidde tot collision met een boom. [?],[?] [?],[?],[?], [?],[?], [?],[?],[?], [?],[?],[?],[?],[?],[?],[?],[?],[?],[?],[?].

2.2.12 Tsjernobyl

[illegible]

2.2.13 oekraine powergrid

Op 23,december 2015 vind er een cyber aanval plaats op het elektriciteitsnet van de Oekraïne. Dit was de eerste bekende aanval op een elektrisch controle systeem. Verschillende onderzoeken zijn gedaan naar de Ukraine cyber aanval, inclusief hoe de actoren zich zelf toegang gaven tot het controle systeem, welke methoden de actoren hebben gebruikt voor reconnaissance en vastleggen van het systeem, een gedetailleerde omschrijving van de aanval op 15 December 2015, en de methoden die gebruikt zijn door de aanvallers om hun sporen uit te wissen en daarmee het stoppen van schade toebrengen nog moeilijker maken. Daarnaast wordt er een gedetailleerde omschrijving gegeven van de beveiliging van de SCADA control systemen gebaseerd op best practices, inclusief het control network ontwerp, technieken voor whitelisting, monitoring en loggen, en opleiding van personeel. [?] [?] [?] [?] [?] [?] [?] [?],[?],[?],[?],[?],[?],[?],[?]. Uit onderzoek[?] naar de aanval, uitgevoerd door Oekraïense en Amerikaanse militairen blijkt bleek onder meer dat de power grids in sommige gevallen beter waren beveiligd dan de Amerikaanse. Desondanks was de veiligheid niet optimaal door onder andere de hetgegeven dat werknemers op afstand konden inloggen en geen gebruik van 2-stapsverificatie. Oekraïne wijst naar de russen [?] [?] [?] [?] [?] [?] [?]. De Oekraïne is op het moment van de aanval al in oorlog met Rusland

[?], [?]. [?], [?], [?]. [?] Uit onderzoek bleek al snel dat het om een gecoördineerde aanval ging waarbij statelijke actoren waren betrokken. [?], [?], [?], [?]. Voor de aanval werd gebruik gemaakt van malware verspreid door phishing mails [?], [?]. Uitvoering van de aanval 1. Een eerste spearphishing-aanval per e-mail lokt ontvangers om een bijgevoegd Microsoft®-document te openen met een macro waarop Black Energy 3 (BE3) wordt geïnstalleerd op de werkstations. 2. BE3 en andere tools voeren verkenningen uit en creëren een backdoor in het netwerk. 3. Als gevolg van netwerkverkenning kunnen hackers en krijgen toegang tot Microsoft van de oblenergos Active Directory®-servers die zakelijke gebruikers bevatten accounts en inloggegevens. 4. Met de verzamelde inloggegevens die de kwaadwillende hackers gebruiken een gecodeerde tunnel van een extern netwerk om toegang te krijgen binnen het oblenergo-netwerk. 5. Hackers actoren ontdekken en krijgen toegang tot het controlecentrum toezichtcontrole en data-acquisitie (SCADA) human-machine interface (HMI)-servers en de onderstations waarbij ook de firewallregels worden geconfigureerd. 6. Op 23 december 2015 om 15.30 uur proberen de kwaadwillige hackers de gecompromitteerde SCADA-werkstations over te nemen. Hackers nemen de controle over van HMI-operatoren. 7. Er worden verschillende acties uitgevoerd A. Lancering van een gecoördineerde Telefonie Denial of Service (TDoS)-aanval B. Het uitschakelen van de UPS'en voor de controlecentra. C. Corruptie van de firmware op een externe terminaleenheid (RTU) HMI-module en serieel-naar-Ethernet-poort servers. 8. Kwaadwillige actoren voeren KillDisk-malware uit in een proberen de HMI's van het controlecentrum en de draaipuntwerkstations weg te vagen. [?], [?]. Er zijn wel maatregelen te nemen tegen dergelijke aanvallen waarbij gebruik wordt gemaakt van [?] [?] spearfishing, blackenergy remote access capabilities, serial-to-ethernet communication devices, telephony denial of service attacks. Enkele oplossingen zijn:

1. Identificeer alle risico's en schrijf een plan voor het managen van de risico's.
2. Implementeer effectieve controle om het risico te managen.
3. Creëer een diepgaand model dat ervoor zorgt dat er effectieve en efficiënte security controls worden uitgevoerd.
4. Aangaande de gebeurtenissen in de oekraïne kunnen de volgende security controls worden opgenomen in het securitymodel: Initial access to enterprise network, pivot in enterprise network, elevate privileges, maintenance access, gain access to control system, attack, attack complication, destroy hard drives.

[?] [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?].

2.2.14 Stint

stint ongeluk Vier kinderen, een bestuurder kwamen om en een vijfde persoon, een kind raakte zwaar gewond. Uit onderzoek van bleek: Foutieve toetsing voor de gashendel werd geleverd Geen van de drie onderzochte voertuigen haalden de wettelijk vereiste remvertraging De automatische parkeerrem kan leiden tot gevaarlijke situaties wanneer deze ongewenst geactiveerd wordt tijdens het rijden. Het losraken van de nuldraad naar de gashendel leidt volgens TNO tot ongewenst versnellen van het voertuig en een oncontroleerbare situatie voor de bestuurder. Voor alle drie onderzochte voertuigen geldt dat het ontbreken van een zitplaats leidt tot veiligheidsrisico's voor remmen en sturen door de grotere kans dat de bestuurder van het voertuig valt. Als de bestuurder van een Stint valt, leidt dit in alle rij situaties tot een onbeheersbare situatie [?]

2.2.15 Safety critical systems

De kennis van de mens is verantwoordelijk voor het scheppen van kennis. Opdoen van kennis van veiligheidsintegriteit ten overstaan van onwetendheid. Kennisdoelen zijn daarom altijd belangrijk en kunnen worden onderscheiden in 3 domeinen: cognitief, affectief en psychomotorisch.[?] Een Veiligheidsanalyse is een manier voor het evalueren van ongelukken en risico's op verschillende niveaus. Met een veiligheidsanalyse wordt gekeken naar mensen en systemen die worden blootgesteld aan een gevaar/risico en de mogelijkheid deze te minimaliseren. Veiligheid is moeilijk te definiëren omdat het een vaag concept is. Een veiligheidsanalist moet kennis hebben van het systeem, maar ook ervaring met het systeem en vooral hoe een fout in het systeem zich voordoet. Volgens Stoney[?] is veiligheid een aspect dat de veiligheid van mensenleven of de omgeving niet in gevaar brengt. De interactie tussen systeem en omgeving levert kennis op door ad-hoc ervaring. Maar ook is kennis van abstractie en systeemdoelen belangrijk. Het kwalificeren van informatie moet ook op waarde geschat worden. Een verkeerde schatting kan fataal zijn. Zelfs als een veiligheidsanalyse is gedaan waarbij risico's zijn gedefinieerd en geprioritiseerd is een definitie van een veiligheidsniveau moeilijk. Het zijn hier richtlijnen voor en soms ook niet. Er zijn voorbeelden van tools en technieken te gebruiken voor een veiligheidsanalyse. En dat zijn

1. checklists voor critical safety items
2. fault tree analysis
3. event tree analysis
4. failure modes en critical effects analysis (FMECA, FMET)
5. hazard operability studies

[?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?].

Onderzoeksresultaten naar sluisbeveiliging Een belangrijk onderdeel van de besturing van sluizen is veiligheid. Veiligheid is een breed begrip. Zo kan bij veiligheid worden gekeken naar de grootte van de sluis en de sluis kamers, het intake en discharge systeem, gebruikte materialen bij de bouw, bodemveiligheid, bescherming tegen hoogwater, wind[?, 2-3] en droogte. Maar ook de hydrauliek van de sluisdeuren en mechanische constructies moeten getoetst worden op veiligheid. Kunnen de onderdelen van de sluis omgaan met zout- en zoutwater[?, 2-9]. Daarnaast kun je bij veiligheid denken aan de maatregelen genomen bij onderhoudswerkzaamheden en reconstructies. Maar bij het dagelijks gebruik van sluizen spelen ook sluiscontrole en de daarbij behorende communicatiekabels[?, 2-11], operationaliteit en weggebruikers[?, 2-13] een rol. Enkele voorbeelden van standaarden zijn terug te vinden op pagina[?, 2-43] en [?, 2-54] van het rapport Verouderde computersystemen zijn door de jaren heen gekoppeld aan netwerken, zodat ze op afstand te besturen zijn. Dit zorgt ervoor dat systemen kwetsbaar zijn voor aanvallen van buitenaf. De beveiliging is in de loop der jaren niet voldoende ontwikkeld om de infrastructuur goed te beveiligen. Uit onderzoek[?] blijkt dat de afgelopen jaren wel het nodige geïnvesteerd om de beveiliging op te schroeven, maar deze maatregelen zijn nog onvoldoende doorgevoerd. Uit het rapport Digitale dijkverzekering: cybersecurity en vitale waterwerken blijkt dat crisisdocumentatie is verouderd en er worden geen voldoende pentesten uitgevoerd. Uit het onderzoek blijkt dat nog niet alle vitale waterwerken rechtstreeks zijn aangesloten op het Security Operations Center (SOC) van Rijkswaterstaat. Hierdoor bestaat het risico dat RWS een cyberaanval niet of te laat detecteert. De minister van Infrastructuur en Waterstaat moet nog stappen zetten om aan de eigen doelstellingen voor cybersecurity te voldoen. De Algemene Rekenkamer beveelt de minister van Infrastructuur en Waterstaat

ook aan om het actuele dreigingsniveau te onderzoeken en te besluiten of extra mensen en middelen nodig zijn. Ook is het voor een snelle en adequate reactie op een crisissituatie van essentieel belang dat informatie up-to-date is. Pentesten zouden integraal onderdeel uit moeten maken van de cybersecuritymaatregelen bij vitale waterwerken. Verder zou moeten worden bezien of medewerkers van het SOC beter moeten worden gescreend[?]. Ook het crisismodel kan beter, is de derde deelconclusie van de Algemene Rekenkamer. Er is geen specifiek scenario voor een crisis die wordt veroorzaakt door een cyberaanval. Daarnaast ontbreekt inzicht in de effecten van een cybercrisis op andere sectoren, de zogeheten cascade-effecten. Tevens is de crisisdocumentatie op onderdelen verouderd[?]. Uit een andere conclusie stellen de onderzoekers dat cyberveiligheid nog geen volwaardig onderdeel uitmaakt van reguliere inspecties.' De Rekenkamer hamert erop dat alle vitale waterinfrastructuur zo snel mogelijk op het SOC wordt aangesloten. Ook zouden werknemers van Rijkswaterstaat die belangrijke waterkeringen bedienen beter gescreend moeten worden op hun antecedenten. Sollicitanten hoeven nu slechts een Verklaring Omtrent Gedrag te overleggen, maar dat is een heel lichte toets [?]. Volgens Rijkswaterstaat[?] is het kostbaar en technisch uitdagend om klassieke automatiseringssystemen te moderniseren en wordt er daarom vooral ingezet op detectie van aanvallen en een adequate reactie daarop. Uit het onderzoek blijkt dat Rijkswaterstaat de afgelopen jaren zelf van alle tunnels, bruggen, sluizen et cetera heeft vastgesteld welke cyberveiligheidsmaatregelen moeten worden genomen. Een groot deel van die maatregelen (ongeveer 60 procent) was begin 2018 ook al uitgevoerd, maar Rijkswaterstaat ziet onvoldoende toe op de uitvoering van het resterend deel en heeft geen actueel overzicht van de overgebleven maatregelen. De minister heeft een aantal waterwerken die Rijkswaterstaat beheert als vitaal aangewezen. . Uit het onderzoek blijkt dat nog niet alle vitale waterwerken rechtstreeks zijn aangesloten op het Security Operations Center (SOC) van Rijkswaterstaat. De ambitie om eind 2017 bij alle vitale waterwerken cyberaanvallen direct te kunnen detecteren was in het najaar van 2018 daarmee nog niet gerealiseerd. Hierdoor bestaat het risico dat RWS een cyberaanval niet of te laat detecteert[?]. Over de cyberbeveiliging van gemeenten en waterschappen wordt al langer geklaagd. Zo meldde onderzoeksprogramma voor de televisie EenVandaag al in 2012 dat rioolgemalen en sluizen gemakkelijk van afstand te bedienen waren, onder meer door bijzonder slechte wachtwoorden [?]. Toch zijn er bedrijven die naar oplossingen zoeken. Zo doet het bedrijf Rittal onderzoek naar afstand bediendbare sluizen[?].

Dan is er nog de Beveiligde VPN oplossing. M2M Services levert aan inmiddels 220 gemeenten en waterschappen beveiligde connectiviteitsoplossingen voor het beheer van pompen, riolen en gemalen. Om risico's op beveiligingsincidenten te voorkomen maakt het bedrijf gebruik van een VPN oplossing, waarbij de verbinding optimaal beveiligd is middels encryptie en authenticatie[?]. Veiligheid op het water én op het land Gebruik van lampbewaking [?].

Afbakening van requirements Wet en regelgeving voor sluizen Omdat we in dit onderzoek uitgaan van het uitbreiden van bestaande sluizen is er literatuurstudie gedaan naar sluizen. In de archieven van het ministerie van verkeer en waterstaat is er het rapport Design of waterlocks[?]. Het programma van requirements kunnen we in ons model niet helemaal overnemen. Zo zijn er precondities zoals topografie, bestaande watersluizen, waterlevel, wind, morfologie en bodemeigenschappen.

3 Logica

3.1 Propositielogica

In de propositielogica onderzoekt men het waarheidsgehalte van samengestelde uitspraken aan de hand van elementaire proposities en logische voegwoorden. Wij noemen de woorden 'en', 'of', 'als . . . dan . . .', 'impliceert' logische voegwoorden. Hoewel het taalkundig geen voegwoorden zijn, noemen wij de ontkenningen 'niet' en 'geen' toch logische voegwoorden.

In de propositielogica worden proposities gemaakt van elementaire proposities en logische voegwoorden. In de symbolische propositielogica, de propositierekening, wordt een propositieformule gemaakt van variabelen (letters), logische operatoren en haakjes. Net zoals een formule in de rekenkunde, heeft het deel van een propositieformule binnen de haakjes een hogere prioriteit dan het deel buiten de haakjes. Bovendien wordt alles wat tussen haakjes staat, beschouwd als een eenheid. Om een propositieformule correct samen te stellen, moeten wij verplicht gebruik maken van de volgende grammaticale regels:

1. Een variabele is een formule;
2. Als p een formule is, dan is $\neg p$ een formule;
3. Als p en q formules zijn, dan zijn $(p \wedge q)$, $(p \vee q)$, $(p \rightarrow q)$ en $(p \leftrightarrow q)$ formules.

In een propositieformule mogen haakjes worden weggelaten mits er geen verwarring ontstaat

Tijdens het logisch beschrijven van logica of bij het maken van formules over formules, is het mogelijk dat wij een paradoxale uitspraak doen. Zo'n uitspraak is vergelijkbaar met de volgende zin: Deze zin is onwaar. De bovenstaande zin beschrijft zichzelf. Is deze zin nu waar of onwaar? Dit is een paradox. Wij moeten de uitspraken over de logica scheiden van de logica zelf. Daarom introduceren wij metasymbolen. Deze metasymbolen zijn geen onderdeel van de beschreven logica, maar een toevoeging aan de omgangstaal in dit dictaat.

Voor het bepalen van de waarheidswaarde van een formule $f(p_1; p_2; \dots; p_n)$ moeten alle mogelijke combinaties van waardetoekenningen worden bepaald. Deze combinatie van waardetoekenningen vormen samen de waarheidstabel van deze formule.

3.2 Predicatenlogica

Hoewel de volgende redentatie in de propositielogica ongeldig is, wordt zij intuïtief toch als geldig beschouwd:

1. f_1 "alle republieken plegen geen overspel"
2. f_2 "sommige overspeligen zijn president"
3. y "sommige presidenten zijn geen republieken"

De geldigheid van deze redentatie is gebaseerd op informatie, waarmee de propositielogica geen rekening mee houdt. Om deze extra informatie bij het redeneren te betrekken, moeten wij de propositielogica uitbreiden met de begrippen eigenschappen, variabelen en kwantoren. De zin "alle mensen zijn sterfelijk" geeft aan dat objecten, die de eigenschap 'menselijk zijn' hebben, blijkbare ook de eigenschap 'sterfelijk zijn' hebben. uitspraak symbolisch:

1. sterfelijk objecten $S(x)$

2. menselijke objecten $M(x)$
3. x is een priemgetal $P(x)$

In de uitspraken geven wij de eigenschappen sterfelijk en priemgetal aan met de hoofdletters S en P . De variabele x stelt objecten voor. Een variabele, waarvan de waarde onbekend is, noemen wij vrije variabele. Definitie 4.1 (Predikaat) Een predikaat is een uitspraak met vrije variabelen, die een propositie wordt zodra alle vrije variabelen in dat predikaat gebonden zijn aan een waarde.

4 Computation tree logic

4.1 Algemeen

Doel van de studie is het modelleren van een geautomatiseerde sluis in Upaal. Het testen van het model op correctheid, reliability, safety en liveness kan worden uitgewerkt aan de hand van specificaties en proposities vertaald uit de requirementsanalyse. De kripke structuur is een set van locaties, transities, guards, klokken en data-variabelen waarmee een real-time model van een geautomatiseerde sluis kan worden vertaald naar een wiskundig model.

4.2 De Kripke structuur

Een getimede automatisering is een 6-tupel $A = (\Sigma, S, S_0, X, I, T)$ zodat:

Σ is een eindige set

S is een eindige reeks locaties

$S_0 \subseteq S$ is een set startlocaties

X is een set klokken

$I : S \rightarrow C(X)$ is een mapping van locaties naar klokbeperkingen, de zogenaamde locatie-invariant.

$T \subseteq S \times \Sigma \times C(X) \times 2^X \times S$ is een reeks overgangen. De 5-tupel $\langle s, a, \varphi, \lambda, s' \rangle$ komt overeen met een overgang van locatie s naar locatie s' gelabeld met a , een beperking φ die specificeert wanneer de overgang wordt ingeschakeld, en een set klokken $\lambda \subseteq X$ die worden gereset wanneer de overgang wordt uitgevoerd.

Een model voor een getimede automatisering A is een oneindige toestandsovergangsgrafiek $\tau(A) = (\Sigma, Q, Q_0, R)$. Elke toestand in Q is een paar (s, v) waarbij $s \in S$ een locatie is en $v : X \rightarrow \mathbb{R}^+$ een kloktoewijzing is, waarbij elke klok wordt toegewezen aan een niet-negatieve reële waarde. De reeks begintoestanden Q_0 wordt gegeven door $(s, v) \in Q_0 \iff s \in S_0 \wedge \forall x \in X [v(x) = 0]$.

Om de toestandstransitierelatie voor $\tau(A)$ te definiëren, moeten we eerst een notatie invoeren. Voor $\lambda \subseteq X$ definieert u $v[\lambda := 0]$ als de kloktoewijzing die hetzelfde is als v voor klokken in $X - \lambda$ en wijst de klokken in λ toe naar 0. Voor $d \in \mathbb{R}$ definieert u $v + d$ als de kloktoewijzing die elke klok $x \in X$ toewijst aan $v(x) + d$. De kloktoewijzing $v - d$ wordt op dezelfde manier gedefinieerd. Uit de korte discussie in de inleiding weten we dat een getimede automatisering twee basistypen overgangen kent:

Vertragingsovergangen komen overeen met het verstrijken van de tijd tijdens het verblijf op een bepaalde locatie.

lijn We schrijven $(s, v) \xrightarrow{d} (s, v+d)$, waarbij $d \in \mathbb{R}^+$, op voorwaarde dat voor elke $0 \leq e \leq d$ geldt de invariant $I(s)$ voor $v + e$.

Actieovergangen komen overeen met de uitvoering van een overgang vanuit T . We schrijven $(s, v) \xrightarrow{a} (s', v')$, waarbij een $a \in \Sigma$, op voorwaarde dat er is een overgang $\langle s, a, \varphi, \lambda, s' \rangle$ zodanig dat v voldoet aan φ en $v' = [\lambda := 0]$.

4.3 Soorten transitie

Een transitie bestaat uit een unieke bronlocatie, een unieke doellocatie en een guard, d.w.z. een voorwaarde ($g := x \leq c - g$, waarbij $i, j, =, <, >$ de symbolen zijn om de guard te definiëren. een label (dat gebruikt kan worden voor synchronisatie) een subset (mogelijk leeg) van klokken die opnieuw moeten worden ingesteld

een klokwaardering is een functie $v: X \rightarrow \mathbb{R}^+$

$v[Y := 0]$ is de waardering verkregen uit v door het resetten van klokken uit Y :

$$v[Y := 0] = \begin{cases} 1, & 0 \leq x \in Y. \\ 0, & \text{otherwise.} \end{cases}$$

$v + d \equiv$ tijdstroom (d eenheden)

$(v + d)(x) \equiv v(x) + d$

$v \models c$ betekent dat waardering v voldoet aan de beperking c

evaluatie van een klokbeperking ($v \models g$)

1. $v \models g \times i \leq k$ iff $(x) \leq i \leq k$
2. $v \models x \leq k$ iff $(x) \leq k$
3. $v \models g_1 \wedge g_2$ iff $v \models g_1$ and $v \models g_2$

4.4 Tijd

Tijd is dus een belangrijke eigenschap van time-based model checking. Ook hier is dus een onderzoek naar gedaan. [?]

Modelling en transities Voor algemene kennis over modellen, symbol checking en data variabelen is de literatuur geraadpleegd. [?] [?, p. 14] [?, p. 16–20] [?, p. 22–24] [?, p. 27–29] [?, p. 30] [?, p. 32] [?, p. 40–41] [?, p. 46–49] Aspecten die naar voren komen in de literatuur zijn Fairness in Symbolic model checking' [?, p. 68–71] maar ook '6.4 Counterexamples and whitness' [?, p. 71] 'Partitioned Transition relation' [?, p. 79] '9.1 Automata on finite and infinite words' [?, p. 121] '10.2 Independence and invisibility' [?, p. 144–145] en Equivalences and preorders between structures Fair simulation relation' [?, p. 178]. Ook zijn in de literatuur enkele case studies behandeld van modellen en de daarbij behorende model checking [?, p. 197] door middel van Data abstraction en Atomic propositions' [?, p. 199–203].

Clock regions: Timed transitions en clock constraints Er is een literatuuronderzoek gedaan naar clock regio's onder andere in Uppaal. In de geraadpleegde literatuur is meer informatie gegeven over Invariants [?, p. 215–236], Timed Automata' [?, p. 265–268], 'Parallel composition' [?, p. 268], '17.3 Clock regions' [?, p. 274–281] [?], '2.3 Semantics' pp. 6–7 [?, p. 6–7], 'UNiformlyPriced Timed

Automata' pp. 5-6 [?, p. 5-6], '2.1 Transition Systems en Timed Trees' pp. 2 [?, p. 2]. Dan ook nog Network automata' [?, p. 3] [?, p. 2], en Timed input-output transition systems [?, p. 2], in Uppaal Timed automata [?, p. 3] [?, p. 3]. 'The committed Uppaal Model' [?], 'en de Formal verification' [?, p. 5] [?] daarvan. 'Chapter 2 Timed Automata with data variables and Uppaal' pp. 2-4 [?, p. 4] [?] [?], '2.2 Clock constraints' [?] '2 A model for hybrid systems' [?]

CTL logica Alle veiligheid en reachability requirements formeel gespecificeerd in hoofdstuk ... zijn geverifieerd in uppaal met gebruik an A en E state formulae. Deze zijn als volgt: \sim , ξ , \cong , Δ = or equal by definition, \oplus

Om aan te tonen dat de gedefinieerde specificaties altijd geldig zijn moet de basis specificatie inductief worden opgelost. Meer hierover is terug te vinden in [?] [?, p. 73] [?, p. 82] [?, p. 83] [?, p. 90] [?, p. 91] [?, p. 92] [?, p. 93] [?, p. 98] [?, p. 104] [?, p. 156] [?, p. 197] [?, p. 225] [?, p. 236] [?, p. 315] [?, p. 317] en [?, p. 318]

Timed automata Getimedede automaten [4] [57] dienen hierbij om getimedede systemen te modelleren. Dit zijn eindige-toestandsautomaten uitgerust met klokken die worden gebruikt om beperkingen op te geven aan de hoeveelheid tijd die kan verstrijken tussen twee gebeurtenissen (blz 46). Getimedede kripke-structuren (blz 63) (blz 69) (blz 78) blz 99. [?]

4.5 Guards en invarianten

Urgent locations Is hetzelfde als het toevoegen van een clock x , met een invariant $x_i=0$ op de locatie. Zolang een systeem in een urgente locatie zit mag er geen tijd verstrijken. Bijvoorbeeld als een sluis klaar is en er geen schepen in de sluis aanwezig zijn. Dan moet er een urgentie zijn dat alle schepen waar mogelijk worden opgesteld voor invaren. Als er geen schepen in de wachtrij en er staan geenschepen klaar om in te varen dan is er misschien urgentie om aan de andere kant schepen op te halen.

Committed locations Als een of meerdere locaties ingesteld zijn als committed. Een committed state kan niet vertragen en de volgende transitie moet een transitie zijn waarin de uitgaande edge komt van een committed edge

4.6 Safety

In navolging van L. Lamport, die stelt dat er bij een veiligheidseigenschap nooit iets ergs mag gebeuren. Het "slechte" vertegenwoordigt een kritieke systeemtoestand die nooit mag optreden. Een Booleaanse waarneembare C nemen: $Tijd \rightarrow 0, 1$, waarbij $C(t) = 1$ dat uitdrukt optijd t het systeem zich in de kritieke toestand bevindt, kan deze veiligheidseigenschap aanwezig zijn uitgedrukt door de formule: $\forall t \in Tijd \neg C(t)$

Hier is $C(t)$ de afkorting van $C(t) = 1$ en dus geeft $\neg C(t)$ aan dat op dat moment t het systeem zich niet in de kritieke toestand bevindt. Dus voor alle tijdstippen t Het is niet zo dat het systeem zich in een kritieke toestand bevindt. Over het algemeen wordt een veiligheidseigenschap gekarakteriseerd als een eigenschap die binnen een bepaalde tijd kan worden vervalst. In het geval van (1.1) een enkele vertonen tijdstip t_0 met $C(t_0)$ volstaat om aan te tonen dat (1.1) niet geldt. In het voorbeeld is een

oversteekplaats met permanent gesloten poorten veilig, maar voor de wachtende auto's en voetgangers is het onaanvaardbaar. Daarom we hebben andere soorten eigenschappen nodig. liveness properties Veiligheidseigenschappen geven aan wat er wel of niet mag voorkomen, maar eis niet dat er ooit iets gebeurt. Levendigheidseigenschappen geven aan wat er moet gebeuren. De eenvoudigste vorm van een liveness-eigenschap garandeert dat er uiteindelijk iets goeds gebeurt. De 'goede zaak' vertegenwoordigt een wenselijke systeemtoestand, bijvoorbeeld de poorten open voor het wegverkeer. Een Booleaanse waarneembare nemen $G : \text{Tijd} \rightarrow 0, 1$, waarbij $G(t) = 1$ drukt uit dat op tijdstip t de systeem in goede staat verkeert, kan deze levendigheidseigenschap tot uitdrukking worden gebracht volgens de formule: $t \in \text{Tijd} \vdash G(t)$. Met andere woorden: er bestaat een tijdstip waarop het systeem zich in de goede staat. Houd er rekening mee dat deze eigenschap niet in bounded kan worden vervalst tijd. Als voor enig tijdstip t_0 alleen $\neg G(t)$ is waargenomen $t \leq t_0$, we kunnen niet klagen dat (1.2) uiteindelijk wordt geschonden zegt niet hoe lang het zal duren voordat de goede toestand zich zal voordoen. Een dergelijke liveness-eigenschap is niet sterk genoeg in de context van realtime systemen. Hier zou men graag een tijdgebonden zien wanneer de goede toestand ontstaat. Dit brengt ons bij het volgende soort vastgoed. begrensde responseeigenschappen

Een begrensde responseeigenschap stelt dat een gewenste systeemreactie op een invoer vindt plaats binnen een tijdsinterval $[b, e]$ met ondergrens b Tijd en bovengrens e Tijd waar $b \leq e$. Bijvoorbeeld wanneer een voetganger bij een stoplicht duwt op de knop om over te steken, moet het licht voor voetgangers draaien groen binnen een tijdsinterval van bijvoorbeeld $[10, 15]$. De behoefte aan een bovenwerk grens is duidelijk: de voetganger wil binnen korte tijd de weg oversteken tijd (en niet uiteindelijk). Er is echter ook een ondergrens nodig omdat het verkeerslicht niet ogenblikkelijk van groen naar rood mag gaan, maar pas na een gele fase van bijvoorbeeld 10 seconden om auto's zachtjes afremmen. Waarbij $P(t)$ het indrukken van de knop op tijdstip t en voorstelt $G(t)$ die een groen verkeerslicht voor de voetgangers voorstelt op tijdstip t , we kunnen de gewenste eigenschap uitdrukken met de formule $\forall t_1 \in \text{Tijd} \cdot (P(t_1) \rightarrow \exists t_2 \in [t_1 + 10, t_1 + 15] G(t_2))$ Merk op dat deze eigenschap binnen een bepaalde tijd kan worden vervalst. Wanneer voor een bepaald tijdstip t_1 met $P(t_1)$ ontdekken we dat gedurende de tijd interval $[t_1 + 10, t_1 + 15]$ verscheen geen groen licht voor de voetgangers, eigendom (1.3) wordt geschonden. Duration properties Een duureigenschap is subtieler. Het vereist dat voor observatie-intervallen $[b, e]$ die voldoen aan een bepaalde voorwaarde $A(b, e)$ de geaccumuleerde tijd waarin het systeem zich in een bepaalde kritieke fase bevindt toestand heeft een bovengrens $u(b, e)$. De lekstatus van bijvoorbeeld een gasbrander, waarbij gas ontsnapt zonder dat de vlam brandt, moet voorkomen maximaal 50m de geaccumuleerde tijd t van een kritische toestand $C(t)$ in a te meten gegeven interval $[b, e]$ gebruiken we de integrale notie van wiskundige calculus:

$$\int_b^e C(t) dt$$

Vervolgens kan de duureigenschap worden uitgedrukt door een formule:

$$\forall b, e \in \text{Tijd} \bullet A(b, e) \Rightarrow \int_b^e C(t) dt \leq u(b, e)$$

Andere duration properties Queries voor een time based specificatie in Uppaal worden volgens literatuur [?] gedefinieerd als:

- Het is te allen tijde mogelijk dat een zwakke reeks A met tijdsinterval(s) $[x, y]$ komt voor

- Het is te allen tijde mogelijk dat een zwakke reeks A met tijdsinterval(s) $[x, y]$ dat wel doet niet voorkomen
- Het is te allen tijde mogelijk dat een sterke reeks A met tijdsinterval(s) $[x, y]$ komt voor
- Het is te allen tijde mogelijk dat een element uit verzameling A voorkomt binnen het interval $[x, y]$.
- Het is te allen tijde mogelijk dat alle elementen van verzameling A gelijktijdig voorkomen binnen de interval $[x, y]$
- Het is te allen tijde mogelijk dat alle elementen van verzameling A uitsluitend binnen de verzameling voorkomen interval $[x, y]$
- Het is te allen tijde mogelijk dat een element uit verzameling A nooit voorkomt binnen de interval $[x, y]$.
- Het is te allen tijde mogelijk dat alle elementen van verzameling A nooit tegelijkertijd voorkomen binnen het interval $[x, y]$
- Het is te allen tijde mogelijk dat alle elementen van verzameling A nooit uitsluitend binnenin voorkomen het interval $[x, y]$
- Het is te allen tijde waar dat als er een sterke reeks A met tijdsinterval(s) $[x_1, y_1]$ optreedt dan moet het binnen $[x_2, y_2]$ tijdseenheid(en) gebeuren dat een element uit verzameling B voorkomt
- Het is onvermijdelijk dat als alle elementen van verzameling A gelijktijdig voorkomen binnen de interval $[x_1, y_1]$ dan is het mogelijk dat er op enig moment later een zwakke reeks B ontstaat
- tijdsinterval(s) $[x_2, y_2]$ treedt op
- Het is te allen tijde waar dat alle elementen van verzameling A altijd gelijktijdig voorkomen binnen het interval $[x_1, y_1]$ dan moet het in precies $[z]$ tijdseenheid(en) gebeuren dat alle elementen van verzameling B komen gelijktijdig voor binnen het interval $[x_2, y_2]$

4.7 Fairness

Er wordt een fairness constraint opgelegd aan (de planner van) het systeem dat het proces eerlijk selecteert volgende worden uitgevoerd. Technisch gezien is een fairness constraint een voorwaarde voor uitvoeringen (paden) van het systeem model. Hoewel deze beperkingen geen eigenschappen zijn die moeten worden geverifieerd (het zijn eerder veronderstelde omstandigheden). worden afgedwongen door de implementatie), kunnen ze worden uitgedrukt in temporele logica.

fairness is geen te controleren eigenschap van een systeem. Denk eens aan de manier waarop asynchrone systemen zijn gemodelleerd: als een grafiek waarin elk knooppunt een mondiale systeemtoestand vertegenwoordigt, en elke opvolger van een knooppunt komt overeen met de nieuwe mondiale toestand die wordt bereikt door een bepaald van de processen die een lokale transitie maken. Zo'n model bevat doorgaans geen informatie over hoe vaak een proces wordt uitgevoerd, of hoe het volgende proces wordt uitgevoerd het uit te voeren proces wordt gekozen door de planner; in feite is het model onafhankelijk van de onderliggende besturingssysteem en dus van de planner: planning wordt niet-deterministisch behandeld. [?]

Data variabelen Dat variabelen zijn onder andere: water hoog en laag, en aantal schepen in de queue.

Acties Acties in het model zijn onder andere: invaren, uitvaren, deuren openen en sluiten, nivelleren

4.8 Deadlock

• Deadlock een bereikbare staat kan helemaal geen acties uitvoeren – Deadlock hangt af van de reeks acties die een bereikbare staat niet kan uitvoeren • Om de impasse te behouden moet A niet alleen overschatten wat P kan doen, maar ook wat P weigert

4.9 Zeno gedrag

zeno gedrag: de mogelijkheid dat in een eindige hoeveelheid tijd een oneindig aantal handelingen kan worden verricht. Bijvoorbeeld tijdens het nivelleren Bij het opstellen van schepen Bij het laten wachten van schepen Bij het invaren van schepen

5 Verificatie

We moeten aantonen dat een real-time programma voldoet aan de eisen opgesteld en gespecificeerd.

Een formele verificatie van de gespecificeerde requirements ofwel modeleigenschappen wordt gerealiseerd door deze te vertalen naar de query-language van de symbolic model-checker Uppaal. [?],[?]. Het systeem is gemodelleerd als een netwerk van meerdere timed automata in de vorm van templates: controller, sluis, stoplicht, deur, pomp en schip.

Het bewijs van de modeleigenschappen kan wiskundig worden opgesteld met behulp van kwantoren, zoals:

1. E there Exists a path
2. A in All paths
3. F sometime in the Future
4. G Globally in the future
5. X neXtime

Een pad in de kripke structuur M met de eigenschap dat wordt voldaan aan propositie ϕ van locatie x tot aan locatie y kan worden uitgewerkt als volgnde $\langle \vec{\ell}_0, v_0 \rangle, t_0 \models CF_1 \rightarrow CF_2$ iff \forall path ξ starting in $\langle \vec{\ell}_0, v_0 \rangle, t_0 \forall t \in \text{Time}, \langle \vec{\ell}, v \rangle \in C \bullet t_0 \leq t \wedge \langle \vec{\ell}, v \rangle \in \xi(t) \wedge \langle \vec{\ell}, v \rangle, t \models CF_1$ implies $\langle \vec{\ell}, v \rangle, t \models \forall \diamond CF_2$ [?, p.180]

1. $M, s \models E(\Phi_1 \cup \Phi_2) \text{ s } \pi = s_0, s_1, s_2, \dots \in T_{TM}(s) :$
 $\exists(i \geq 0) [M, s_i \models \Phi \wedge \forall(0 \leq j < i) M, s_j \models \Phi]$
2. $M, s \models A(\Phi_1 \cup \Phi_2) \text{ iff } \forall \pi = s_0, s_1, s_2, \dots \in T_{TM}(s) :$
 $\exists(i \geq 0) [M, s_i \models \Phi \wedge \forall(0 \leq j < i) M, s_j \models \Phi]$

[?, p.23]

Een propositie kan ook worden geschreven als : $AGEF_{[x,y]} \models \Phi_1$
 Of geheel als functie:

$$\forall b,e \in \text{Time} \bullet A(b,e) = \int_b^e C(t) dt \leq u(b,e)$$

overzicht

Tabel 1: System requirements.

Eigenschap	query	non-functional requirement type	Verificatie
Eigenschap	De pomp is alleen actief als de sluisdeuren dicht en stoplichten op rood staan.	Verification	Result
Properties	De deuren staan nooit aan beide kanten open.	Verification	Result
Properties	Voor alle paden geldt dat er maximaal 2 schepen in de sluis zijn.	Verification	Result
Properties	Voor alle paden geldt dat er maximaal 2 schepen in de sluis zijn.	Verification	Result
Properties	Een schip kan niet binnen een cyclus van richting wisselen.	Verification	Result
Properties	In een sluis zitten nooit schepen met een tegengestelde richting.	Verification	Result
Properties	Een deur kan pas open als het stoplicht bij die deur op groen staan.	Verification	Result
Properties	Een deur kan pas sluiten als het schip is binnengevaren.	Verification	Result
Properties	Invaren van een schip duurt max 30 tijdseenheden.	Verification	Result
Properties	De deur sluiten kan binnen 30 tijdseenheden	Verification	Result
Properties	De pomp kan niet meer water bijpompen dan toegestaan op het interval [-10,10].	Verification	Result
Properties	Comments	Verification	Result

6 Conclusie

Op basis van de onderzoeksresultaten uit de literatuurstudie komt naar voren dat veiligheid een aspect is dat moet worden meegenomen bij de requirements engineering proces. Bij de therac waren er diverse problemen: communicatie, doorontwikkeling, controle en toetsing Bij de boeing 737 crashes was het probleem van controle en communicatie naar medewerkers Uit de evaluatie van de china explosion 2015 tianjin komt naar voren dat communicatie, transparantie en veiligheid niet altijd prioriteit hadden bij de lokale autoriteiten Bij de tesla autopilot crashes komen soms onvoldoende onderbouwde ontwerpkeuzes naar voren die niet goed zijn afgewogen tegenover het gedrag van de bestuurder. De ramp in Tsjernobyl toont aan hoe autoriteiten een ramp in de doofpot proberen te stoppen Uit de concepten die zijn ontwikkeld en gevisualiseerd in het hoofdstuk ontwikkelgeschiedenis blijkt dat het implementeren van veiligheid in realsystemen niet volledig is geïmplementeerd. Geprobeerd is om de afhandeling van schepen in te regelen via procedures gebruikt door een maincontroller. Een task scheduler in het concept van 19 mei Een waterlevelsensor en een priorityqueue in templates van 12 juni, 13 juni, 23 juni en 16 juli. Of een error handling template in het concept van 22 mei, 6 juni. Dit resulteerde in de concepten van 19 en 20 september Tijdens de studie is naar voren gekomen dat de moeilijkheid van modelleren in uppaal vooral ligt bij de definitie van de propositie logica voor time-based model checking en dan met name clocks en regions. Was het makkelijk te onderzoeken? Waarom? Wat heb ik geleerd Belangrijkste leerpunt van deze studie is het belang van requirements en specificaties. Een goede requirementsdefinitie is belangrijk voor de ontwikkeling van een model alsook het testen van de specificaties en de evaluatie hiervan.

A Edmud Clarke model checking samenvatting

Before we consider a reachability problem, we show how real-time systems can be modeled as parallel compositions of timed automata [3,5]. We assume an interleaving asynchronous semantics for this operation. Let $A_1 = (\Sigma_1, S_1, \mathcal{S}_0^1, X_1, I_1, T_1)$ and $A_2 = (\Sigma_2, S_2, \mathcal{S}_0^2, X_2, I_2, T_2)$ be two timed automata. Assume that the two automata have disjoint sets of clocks, that is $X_1 \cap X_2 = \emptyset$. Then, the parallel composition of A_1 and A_2 is the timed automaton:

$A_1 \parallel A_2 = (\Sigma \cup \Sigma_2, S_1 \times S_2, \mathcal{S}_0^1 \times \mathcal{S}_0^2, X_1 \cup X_2, I, T)$, where $I(s_1, s_2) = I_1(s_1) \wedge I_2(s_2)$ and the edge relation T is given by the following rules:

1. For $a \in \Sigma_1 \cap \Sigma_2$, if $\langle s_1, a, \varphi, \lambda_1, s_1' \rangle \in T_1$ and $\langle s_2, a, \varphi, \lambda_2, s_2' \rangle \in T_2$ then T will contain the transition $\langle (s_1, s_2), a, \varphi, \lambda_1 \cup \lambda_2, (s_1', s_2') \rangle$
2. For $a \in \Sigma_1 - \Sigma_2$, if $\langle s, a, \varphi, \lambda, s' \rangle \in T_1$ and $t \in S_2$ then T will contain the transition $\langle (s, t), a, \varphi, \lambda, (s', t) \rangle$
3. For $a \in \Sigma_2 - \Sigma_1$, if $\langle s, a, \varphi, \lambda, s' \rangle \in T_2$ and $t \in S_1$ then T will contain the transition $\langle (t, s), a, \varphi, \lambda, (t, s') \rangle$

In the definition of timed automata, we allowed the clock constraints that serve as the invariants of locations and the guards of transitions to contain arbitrary rational constants. We can multiply the constants in each clock constraint by the least common multiple m of the denominators of all the constants to integers. The value of a clock can still be an arbitrary nonnegative real number. Note that applying this transformation can change the clock assignments in the set of reachable states of $T(A)$. Fortunately, this does not cause a major problem. The reachable states of the original automaton can be obtained from the locations of the transformed automaton by applying the inverse transformation, that is, dividing each clock value by m .

For example, let A be a timed automaton with two clocks x_1 and x_2 . Let s be a location in A with an outgoing transition e to some other location. Consider two states (s, v) and (s, v') in $T(A)$ that correspond to location s . Suppose that $v(x_1) \equiv 5.3$, $v(x_2) \equiv 7.5$, $v'(x_1) \equiv 5.5$ and $v'(x_2) \equiv 7.9$. Assume that the guard φ associated with e is $x_1 \geq 8 \wedge x_2 \geq 10$. It is easy to see that if (s, v) eventually satisfies the guard, then so will (s, v') . [?, p 274]

The value of a clock can get arbitrarily large; however, if the clock is never compared to a constant greater than c , then the value of the clock will have no effect on the computation of A once it exceeds c . Suppose, for instance, that the clock x is never compared to a constant greater than 100 in the invariant associated with a location or in the guard of a transition.

Then, based on the behaviour of A , it is impossible to distinguish between x having the value 101 and x having the value 1001. <https://www.cis.upenn.edu/~alur/IC93.pdf> Alur, Courcoubetis, and Dill [7,8] show how to formalize this reasoning. For each clock $x \in X$, let c_x be the largest constant that x is compared with in the invariant of any location or in the guard of any transition. For $t \in \mathbb{R}^+$, let $fr(t)$ be the fractional part of t , and let $[t]$ be the integral part of t . Thus, $t = [t] + fr(t)$. We define an equivalence relation \cong on the set of possible clock assignments as follows: Let v and v' be two clock assignments. Then $v \cong v'$ if and only if three conditions are satisfied:

For all $x \in X$ either $v(x) \geq cx$, and $v'(x) \geq gx$ or $[v(x)] = [v'(x)]$. For all $x, y \in X$ such that $v(x) \leq cx$ and $v(y) \leq cy$, $\text{fr}(v(x)) \leq \text{fr}(v(y))$ if and only if $\text{fr}(v'(x)) \leq \text{fr}(v'(y))$. For all $x \in X$ either $v(x) \leq cx$, $\text{fr}(v(x)) = 0$ if and only if $\text{fr}(v'(x)) = 0$. It is easy to see that \cong does indeed define an equivalence relation. The equivalence classes of \cong are called regions [7,8]. We will write $[v]$ to denote the region which contains the clock assignment v . Each region can be represented by specifying

1. for every clock $x \in X$, once clock constraint from the set $x=c \mid c=0, \dots, cx \cup c-1 \mid x \mid c \mid c=1, \dots, cx \cup x \mid cx$ 2. for every pair of clocks $x, y \in X$ such that $c-1 \mid x \mid c$ and $d-1 \mid y \mid d$ are clock constraints in the first condition, whether $\text{fr}(x)$ is less than, equal to, or greater than $\text{fr}(y)$.

Figure 17.7 which is taken from [8], shows the clock regions for a timed automaton with two clocks x and y where $cx = 2$ and $cy = 1$. In this example, there are a total of 28 regions: 6 corner points, 14 open line segments and 8 open regions.

We will use this observation to show that \cong has finite index and, consequently, that the number of regions is finite. Our proof of this fact is based on the proof given in [8].

Lemma 43 The number of equivalence classes that \cong induces on $C(X)$ is bounded by $|X|! \cdot 2^{|X|} \cdot \prod_{x \in X} (2cx + 2)$ *proof* An equivalence class $[v]$ of \cong can be described by a triple of arrays in the following manner. For each clock $x \in X$, the array α tells which of the intervals $[i, i+1]$ contains the value $v(x)$. Thus, the array α represents the clock assignment v if and only if for each clock $x \in X$, $v(x) \in \alpha(x)$. The number of ways to choose α is $\prod_{x \in X} (2cx + 2)$. [?, p 275]

Let X_a be the set of clocks with nonzero fractional part. The array $\beta: X_a \rightarrow 1, \dots, A_a$ is a permutation of X_a , which gives the ordering of the fractional parts of the clocks in X_a with respect to \leq . Thus the array β represents a clock assignment c if and only if for each pair $x, y \in X_a$, if $\beta(x) < \beta(y)$ then $\text{fr}(v(x)) \leq \text{fr}(v(y))$. For a given α , the number of ways to choose β is bounded by $|X_a|!$ which is bounded by $|X|!$.

The third component γ is a boolean array indexed by X_a that is used to specify which clocks in X_a have the same fractional part. For each clock c , $\gamma(x)$ tells whether or not the fractional part of $v(x)$ equals the fractional part of its predecessor in the array β . Thus the array γ represents a clock assignment v if and only if for each $x \in X$, $\gamma(x)$ equals 0 exactly when there is a clock $\gamma \in X_a$ such that $\beta(\gamma) = \beta(x) + 1$ and $\text{fr}(v(x))$ equals $\text{fr}(v(\gamma))$. The number of ways of choosing γ is bounded by the number of boolean arrays over X_a , which is bounded by $2^{|X|}$. Hence, α encodes the integral parts of the clock assignments, and β with γ encodes the ordering of their fractional parts. It is easy to see that the sets represented by triples are equivalence classes of \cong and that every equivalence class is represented by some triple. The bound given in the statement of the lemma is the product of the bounds associated with α , β , and γ . This completes the proof of the lemma. [?, p 276]

The following properties of the equivalence relation \cong are used in later in this chapter. **Lemma 44** Let v_1 and v_2 be two clock assignments, let φ be a clock constraint, and let $\lambda \subseteq X$ be a set of clocks. 1. if $v_1 \cong v_2$ and t is a nonnegative integer, then $v_1 + t \cong v_2 + t$. 2. if $v_1 \cong v_2$, then $\forall t_1 \in \mathbb{R}^{+} [v_1 + t_1 \cong v_2 + t_1]$ 3. if $v_1 \cong v_2$, then v_1 satisfies φ if and only if v_2 satisfies φ 4. If $v_1 \cong v_2$, then $v_1[\lambda := 0] \cong v_2[\lambda := 0]$ [?, p 277]

Note that the first property may not hold if t is not an integer. For example, $(2.8) \cong (.1, .2)$, but $(.2, .8) + .3$ is not equivalent to $(.1, .2) + .3$. All of the properties except the second are straightforward to prove and will be left to the reader. A proof of the second property is sketched below. The proof is not difficult but it is somewhat tedious. It can be safely skipped when this chapter is read for the first time. [?, p 277]

Proof Assume that $v_1 \cong v_2$. We can assume that $t_1 \geq 0$ because, otherwise, we can simply choose $t_2 = 0$. Let x_1, x_2, \dots, x_n . We can treat v_1 as a vector $v_1 = \langle a_1, \dots, a_n \rangle$, where a_i is the value of clock x_i in v_1 . Similarly, we let $v_2 = \langle b_1, \dots, b_n \rangle$. Since corresponding clocks have the same integer part, we can assume without loss of generality that $0 \leq a_i < 1$ and $0 \leq b_i < 1$. Also, assume that the clock values are sorted into increasing order so that $a_1 \leq a_2 \leq \dots \leq a_n$ and $b_1 \leq b_2 \leq \dots \leq b_n$.

case 1 Assume that the largest element in $v_1 + t_1$ is less than or equal to 1. This case is trivial. We can easily choose t_2 so that $v_1 + t_1 \cong v_2 + t_2$

case 2 Assume that $0 \leq t_1 < 1$. Let the first element of $v_1 + t_1$ that is greater than or equal to 1 be $a_k + t_1$. Choose ϵ so that $\epsilon = 0$ if $a_k + t_1 = 1$ and so that $0 < \epsilon < b_k - b_{k-1} - 1$ if $a_k + t_1 < 1$. Note that $b_{k-1} < b_k = b_{k-1}$, then $a_k = a_{k-1}$ and $a + t_1$ is not the first element of $v_1 + t_1$ that is greater than or equal to 1. We will show that $v_1 + t_1 \cong v_2 + (1 + \epsilon - b_k)$. In order to show this we will split the vectors into two parts. Let

$L_1 = \langle a_1 + t_1, \dots, a_{k-1} + t_1 \rangle$, and $L_2 = \langle b_1 + (1 + \epsilon - b_k), \dots, b_{k-1} + (1 + \epsilon - b_k) \rangle$. In each case it is straightforward to show that. [?, p 277]

1. all of the elements are positive 2. the elements are sorted in increasing order, and 3. all of the elements are less than 1. Because of these conditions it is easy to see that $L_1 \cong L_2$. Similarly, let

$R_1 = \langle a_k + t_1, \dots, a_n + t_1 \rangle$, and $R_2 = \langle b_k + (1 + \epsilon - b_k), \dots, b_n + (1 + \epsilon - b_k) \rangle$

All of the elements in R_1 and R_2 are greater than or equal to 1. The fractional parts are given by $R_1 - 1$ and $R_2 - 1$, respectively. For these vectors it is straightforward to show that

1. all of the elements are nonnegative 2. the elements are sorted in increasing order, and 3. all of the elements are less than 1

Moreover, an element in one vector is 0 if and only if the corresponding element in the other vector is 0. Thus $R_1 - 1 \cong R_2 - 1$. It follows immediately that $R_1 \cong R_2$. It is not difficult to see that the fractional parts of R_2 precede the fractional parts of L_2 . Let $i \geq k$ and $j < k$. Then $b_i + (1 + \epsilon - b_k) - 1 \leq b_j + (1 + \epsilon - b_k)$. This is equivalent to $b_i - b_j \leq 1$, which is obviously true. The same relationship holds for the fractional parts of R_1 and L_1 , that is, $a_i + t_1 - 1 \leq a_j + t_1$.

hence, we obtain $R_1 \cdot L_1 \cong R_2 \cdot L_2$, where \cdot is concatenation of vectors. This shows that for all t_1 with $0 \leq t_1 < 1$, there exists a t_2 such that $v_1 + t_1 \cong v_2 + t_2$ and completes the proof of

case 3 Finally, suppose that $t_1 \geq 1$. Let $t_1' = t_1 - [t_1]$, so that $0 \leq t_1' < 1$. Find t_2 such that $v_1 + t_1' \cong v_2 + t_2$. Then: $v_1 + t_1 + [t_1] \cong v_2 + t_2 + [t_1]$.

If we choose $t_2 = [t_1]$, then we have $v_1 + t_1 \cong v_2 + t_2$ as required. This completes the proof of the second property.

The equivalence relation \cong over clock assignments can be extended to an equivalence relation over the state space of $T(A)$ by requiring that equivalent states have identical locations and equivalent clock assignments: $(s, v) \cong (s', v')$ if and only if $s = s'$ and $v \cong v'$. The key property of the equivalence relation \cong is given by the following lemma [5]: [?, p 278]

Lemma 45 If $v_1 \cong v_2$ and $(s, v_1) \xrightarrow{a} (s', v')$. The transition $\langle s, a, \varphi, \lambda, s' \rangle$ that takes state (s, v_1) to state (s', v_1') corresponds to two transitions of the timed automaton.

Proof Assume that $v_1 \cong v_2$ and $(s, v_1) \models a(s', v'_1)$. The transition $\langle s, a, \varphi, \lambda, s' \rangle$ that takes state (s, v_1) to state (s', v'_1) corresponds to two transitions of the timed automaton:

a delay transition $(s, v_1) \models d_1(s, v_1 + d_1)$ for some $d_1 \geq 0$, and an action transition $(s, v_1 + d_1) \models a(s', v'_1)$ such that $v_1 + d_1$ satisfies φ and $v'_1 = (v_1 + d_1)[\lambda := 0]$.

Since $v_1 \cong v_2$ and v_1 satisfies $I(s)$, v_2 also satisfies $I(s)$. Furthermore, there exists $d_2 \geq 0$ such that $v_1 + d_1 \cong v_2 + d_2$. Since $v_1 + d_1$ satisfies $I(s)$, $v_2 + d_2$ also satisfies $I(s)$. Because the clock constraint $I(s)$ is convex and is satisfied by both v_2 and $v_2 + d_2$, $I(s)$ must be satisfied by $v_2 + e$ for all e such that $0 \leq e \leq d_2$. Consequently, the delay transition $(s, v_2) \models d_2(s, v_2 + d_2)$ is legal.

Since $v_1 + d_1 \cong v_2 + d_2$, both $v_1 + d_1$ and $v_2 + d_2$ must satisfy the clock constraint for the guard φ . Thus, the transition $\langle s, a, \varphi, \lambda, s' \rangle$ must also be enabled in the state $(s, v_2 + d_2)$. Let $v'_2 = (v_2 + d_2)[\lambda := 0]$. Then v'_2 is equivalent to v'_1 . Hence, there is an action transition $(s, v_2 + d_2) \models a(s', v'_2)$. Combining the delay transition with the action transition, we get $(s, v_2) \models a(s', v'_2)$ as required.

As a result of the lemma, we can construct a finite state transition graph that is bisimulation equivalent to the infinite state transition graph $T(A)$. The finite state transition graph is called the region graph of A [7,8] and is denoted by $R(A)$. A region is a pair $(s, [v])$. Since \cong has a finite index, there are only a finite number of regions. The states of the region graph are the regions of A . The construction of $R(A)$ will have the property that whenever (s, v) is a state of $T(A)$, the region $(s, [v])$ where s_0 is an initial state of A and v_0 is a clock assignment that assigns 0 to every clock. The transition relation of $R(A)$ is defined so that bisimulation equivalence is guaranteed. There will be a transition labeled with a from the region $(s, [v])$ to the region $(s', [v'])$ if and only if there are assignments $\omega \in [v]$ and $\omega' \in [v']$ such that (s, ω) can make a transition to (s', ω') [?, p 278]. We summarize the construction of the region graph $R(A)$ below. Let $A = (\sigma, S, S_0, X, I, T)$ be a timed automaton. Then, the states of $R(A)$ have the form $(s, [v])$ where $s \in S$ and $[v]$ is a clock region. The initial states have the form $(s_0, [v])$ where $s_0 \in S_0$ and $v(x)=0$ for all $x \in X$. $R(A)$ has a transition $((s, [v]), a, (s', [v']))$ if and only if $(s, \omega) \models a(s', \omega')$ for some $\omega \in [v]$ and some $\omega' \in [v']$. We can use Lemma 45 to prove bisimulation equivalence.

Theorem 31 We will show that $T(A)$ and $R(A)$ are bisimilar. Define the bisimulation relation B by $(s, v)B(s, [v])$. It is easy to see that the initial state (s_0, v_0) corresponds to the state $(s_0, [v_0])$. Next, we show that for each transition of $T(A)$, there is a corresponding transition of $R(A)$, and vice versa. Suppose first that $(s, v)B(s, [v])$. Suppose on the other hand that $(s, v)B(s, [v])$ and that $(s, v) \models a(s', [v'])$. Then there exist $\omega \in v$ and $\omega' \in v'$ such that (s', v') and $(s, v) \models a(s', v')$. Hence $v' \cong \omega \in v'$, so $[v'] = [v']$. By the definition of B , $(s', v')B(s', [v'])$, it follows that $(s', v')B(s', [v'])$. [?, p 279]

B Ontwikkelgeschiedenis

Tijdens dit project zijn verschillende modellen ontwikkeld. Het doel van de concepten is de realisatie van een realtime geautomatiseerde sluis in Uppaal. De requirements specificeren zich in modules zoals sluis, controlekamer, schepen, wachtrijen, sluisdeuren en sensoren.



(a)



(b)

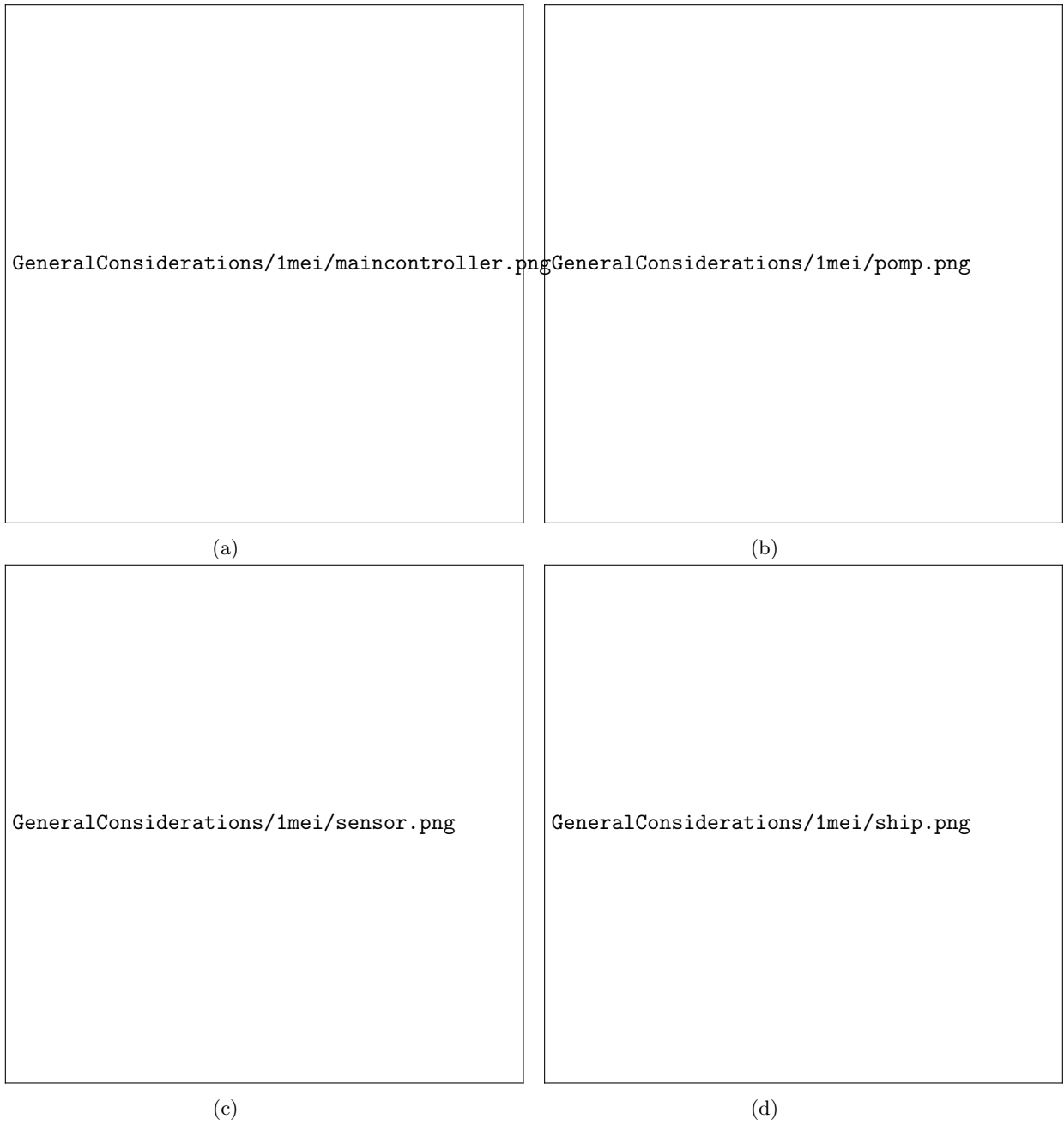


(c)



(d)

Figuur 1: Sluismodel 1 mei

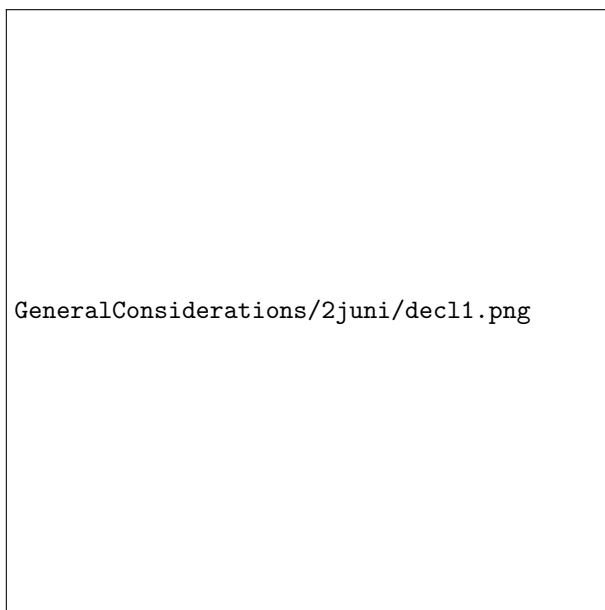


Figuur 2: Sluismodel 1 mei



Figuur 3: Sluismodel 1 mei

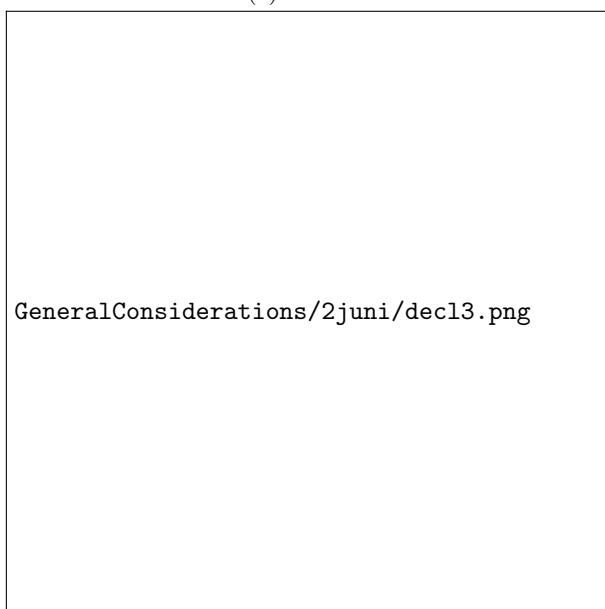
B.1 Ontwerpen: 2 juni



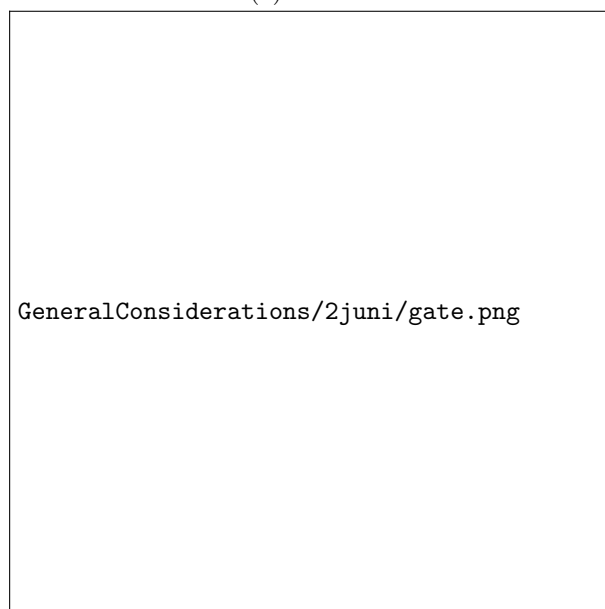
(a)



(b)

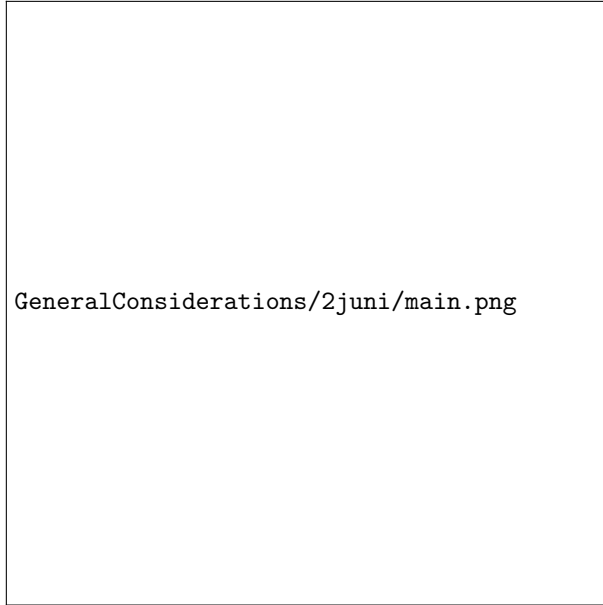


(c)



(d)

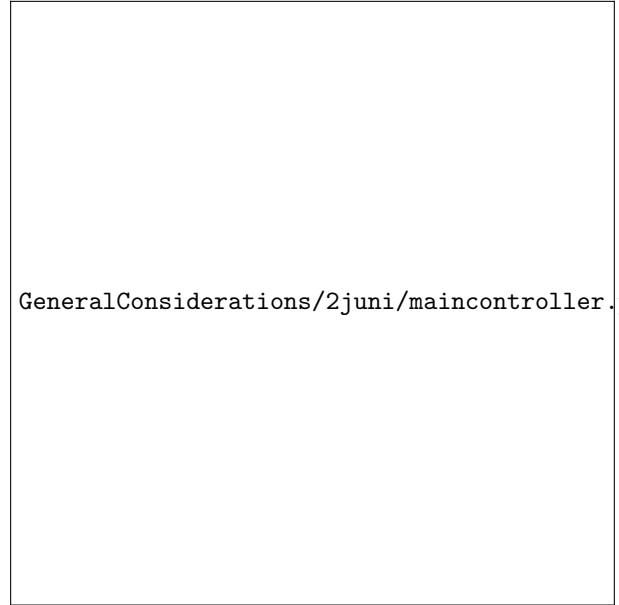
Figuur 4: Sluismodel 2 juni



GeneralConsiderations/2juni/main.png

(a)

(c)



GeneralConsiderations/2juni/maincontroller.png

(b)

(d)

Figuur 5: Sluismodel 2 juni

(a)

(c)

(b)

(d)

Figuur 6: Sluismodel 2 juni

B.2 Ontwerpen: 2 mei

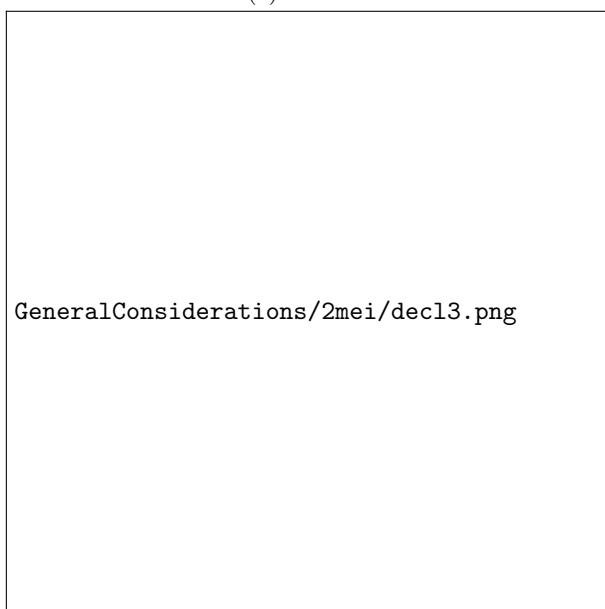
GeneralConsiderations/2mei/wachtrij.png



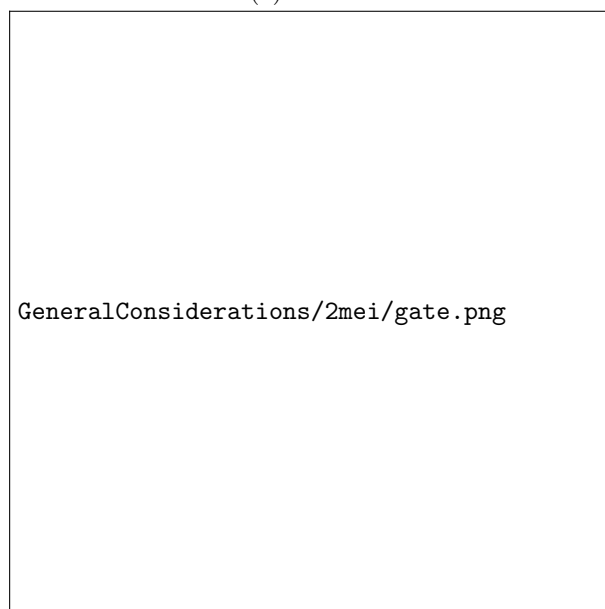
(a)



(b)



(c)



(d)

Figuur 7: Sluismodel 2 mei



(a)



(b)

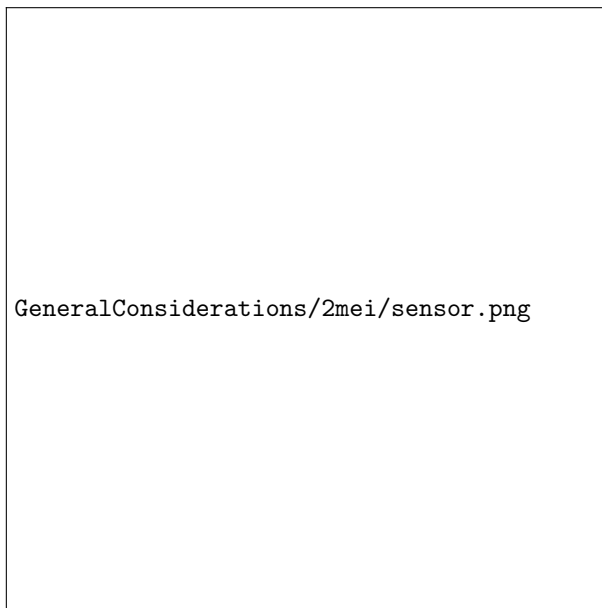


(c)

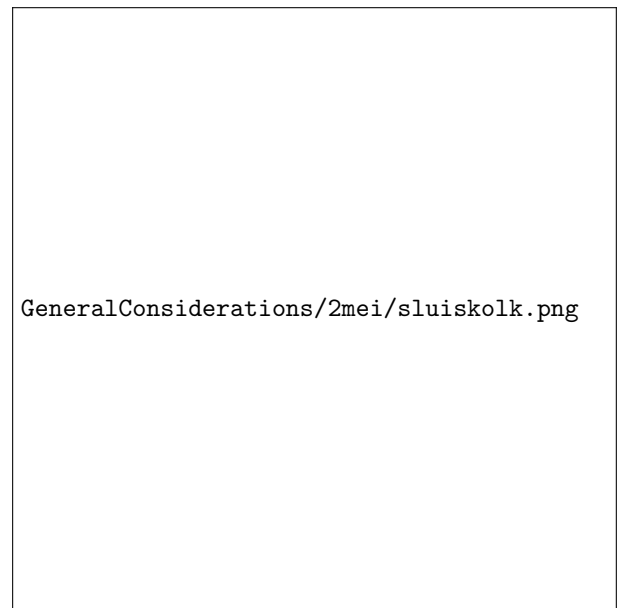


(d)

Figuur 8: Sluismodel 2 mei



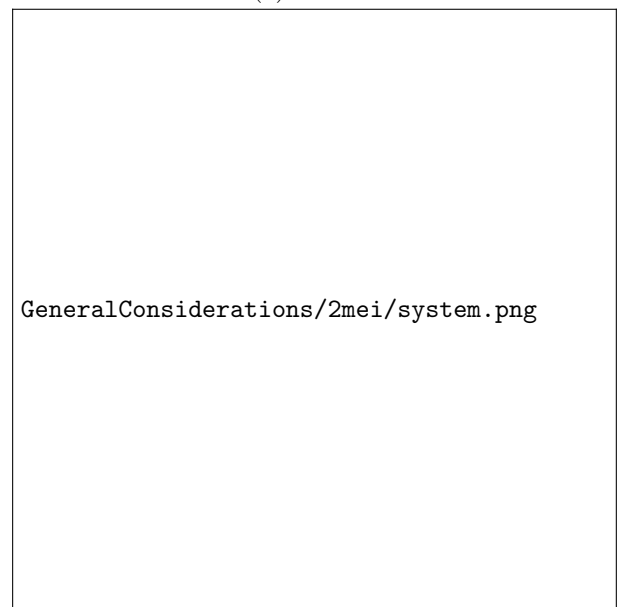
(a)



(b)



(c)



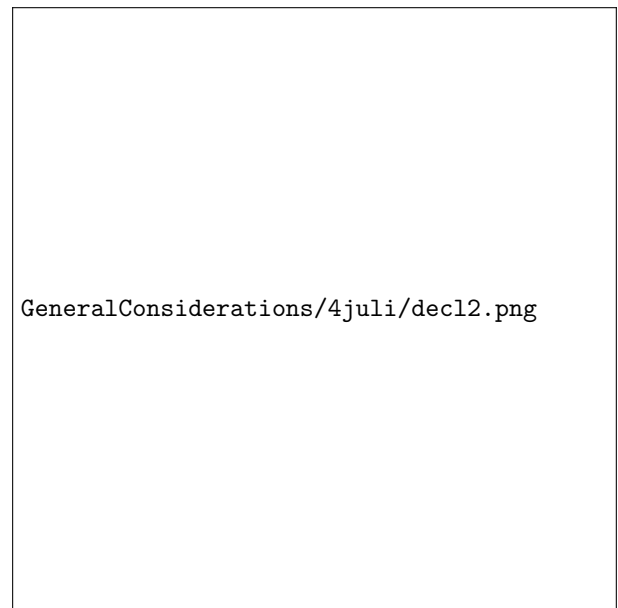
(d)

Figuur 9: Sluismodel 2 mei

B.3 Ontwerpen:4 juli



(a)



(b)



(c)



(d)

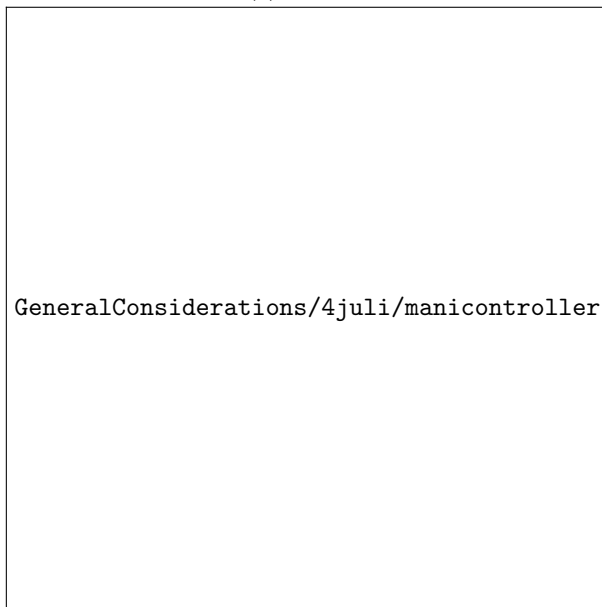
Figuur 10: Sluismodel 4 juli



(a)



(b)



(c)

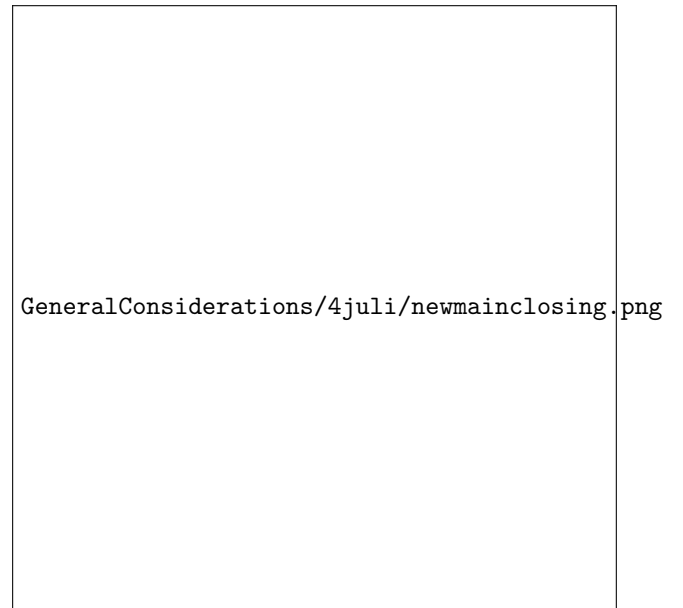


(d)

Figuur 11: Sluismodel 4 juli



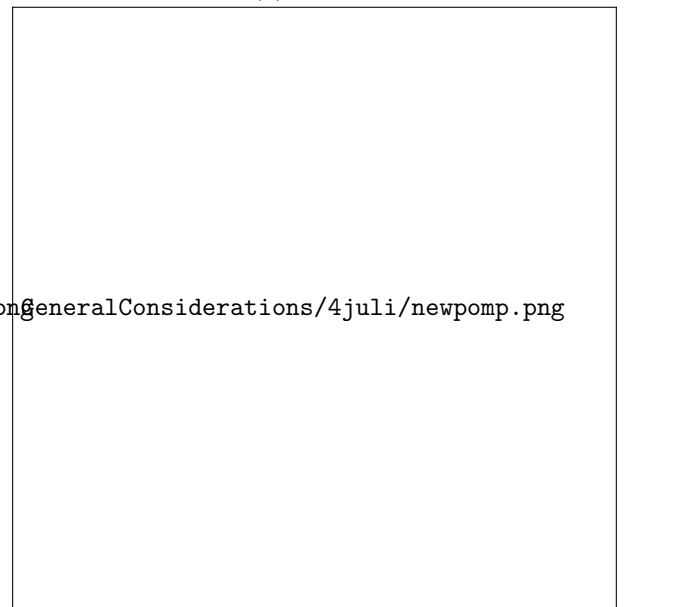
(a)



(b)

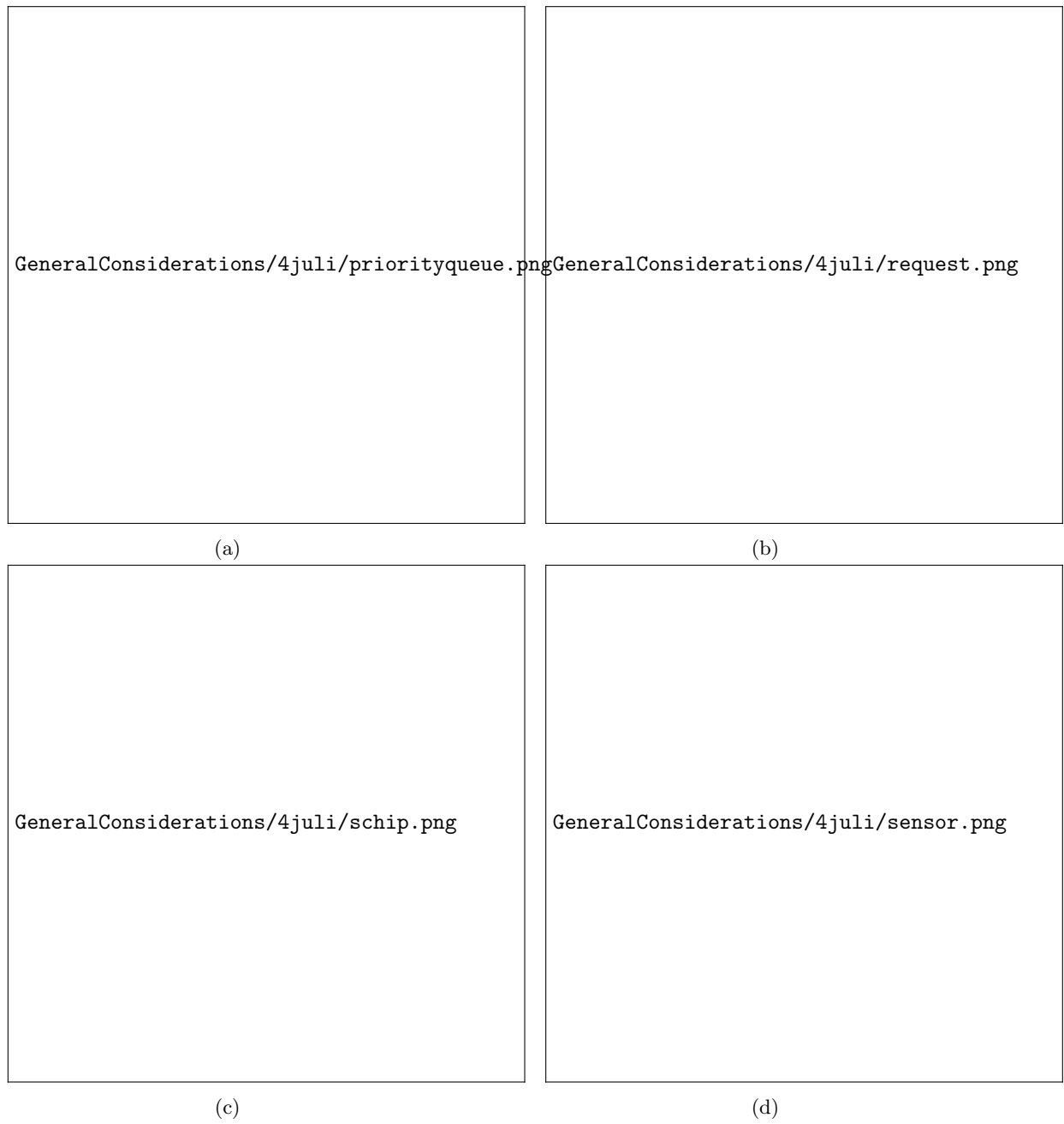


(c)



(d)

Figuur 12: Sluismodel 4 juli



Figuur 13: Sluismodel 4 juli



(a)



(b)



(c)



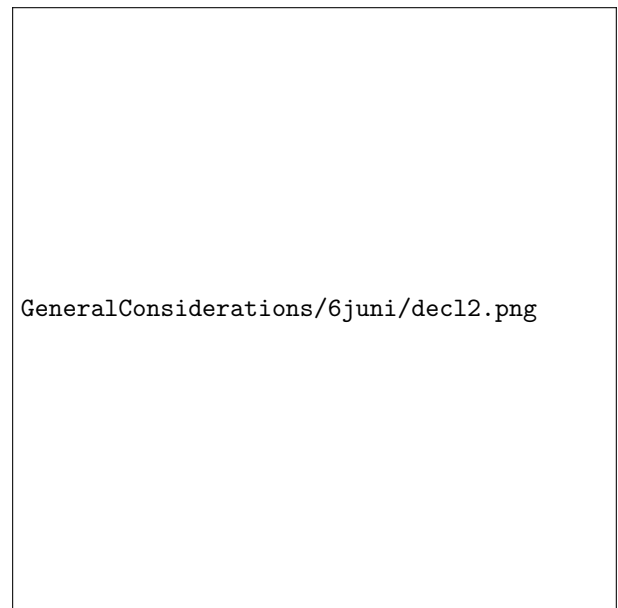
(d)

Figuur 14: Sluismodel van 4 juli

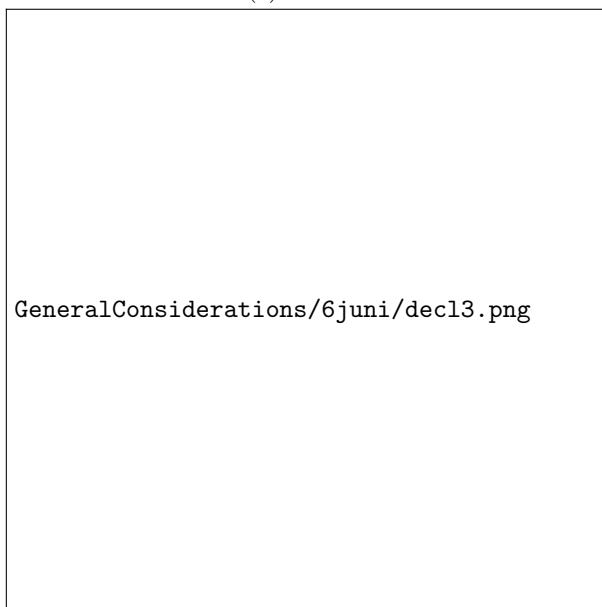
B.4 Ontwerpen: 6 juni



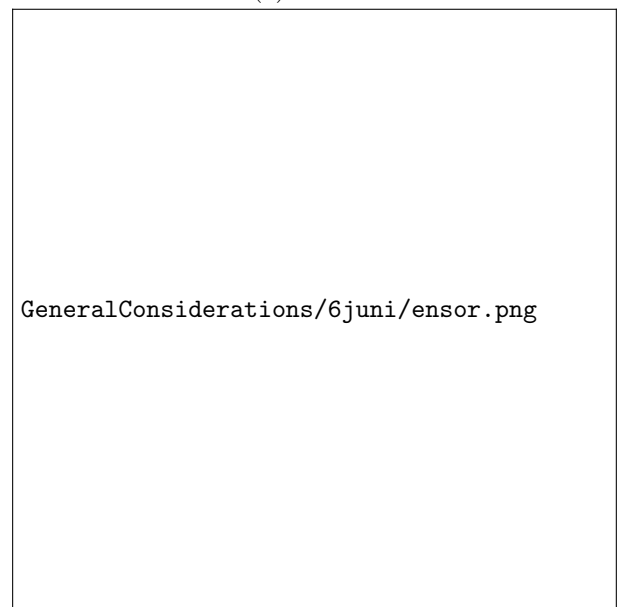
(a)



(b)



(c)



(d)

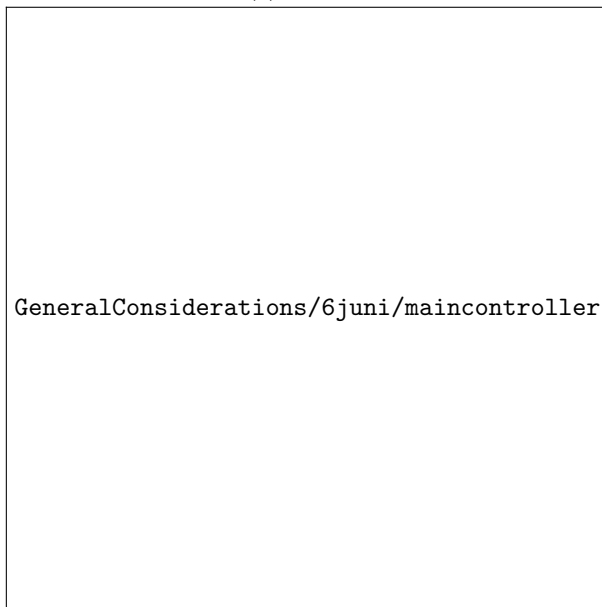
Figuur 15: Sluismodl van 6 juni



(a)



(b)

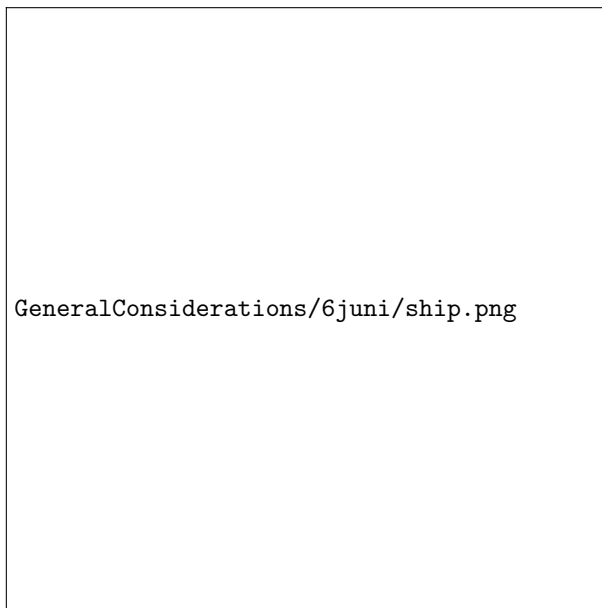


(c)

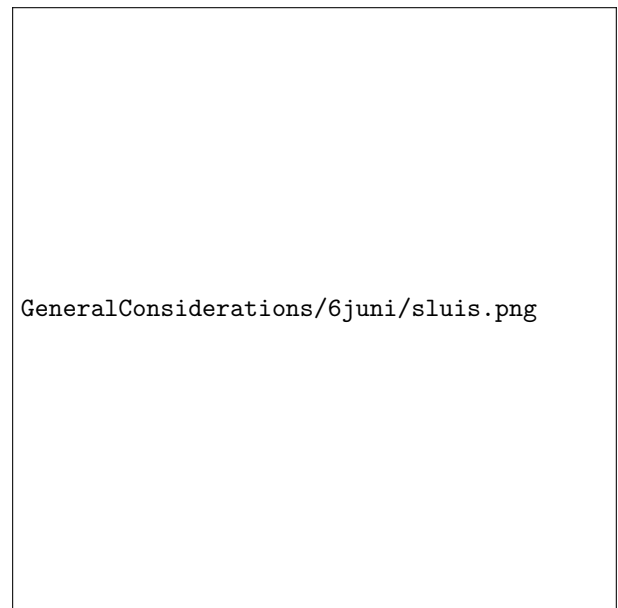


(d)

Figuur 16: Sluismodel 6 juni



(a)



(b)



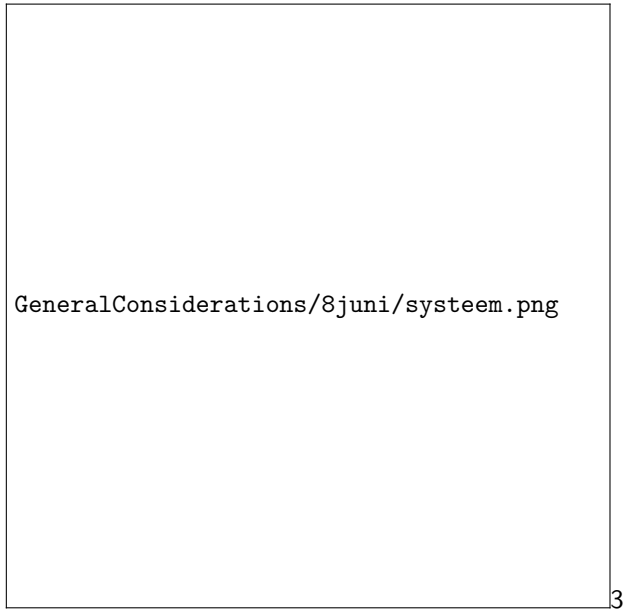
(c)

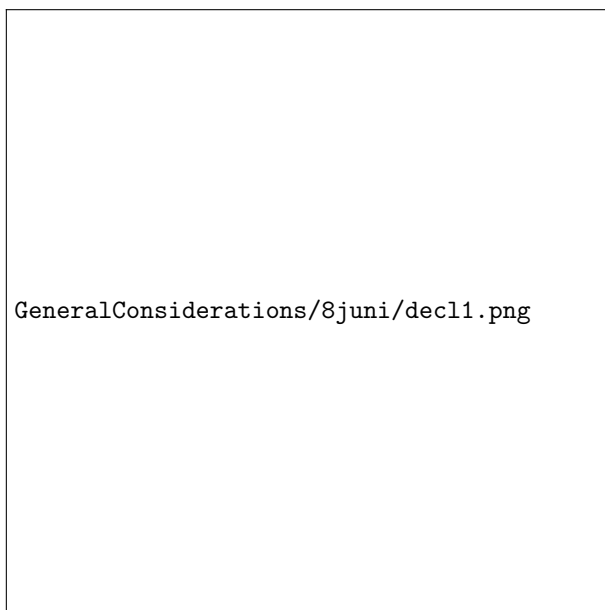


(d)

Figuur 17: Sluismodel 6 juni

B.5 Ontwerpen: 8 juni

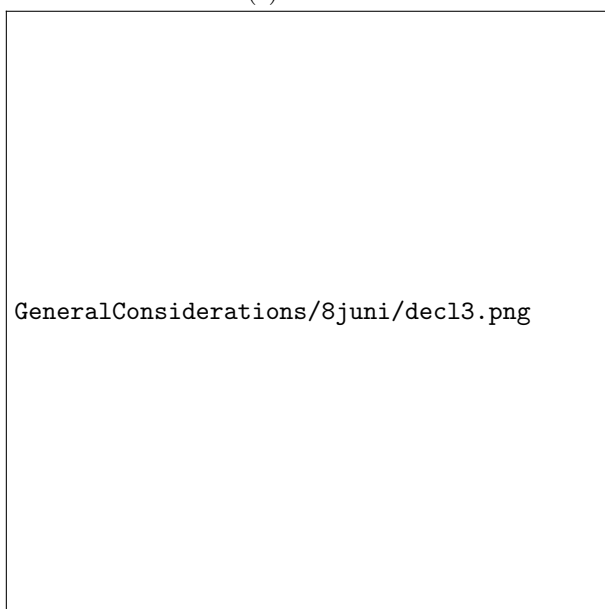




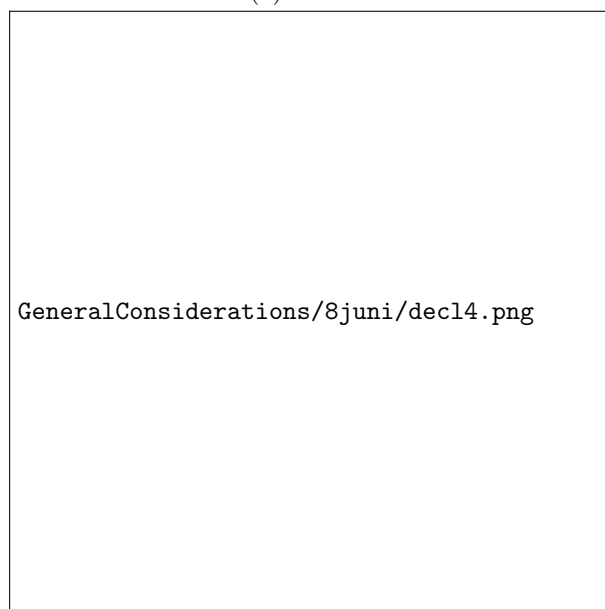
(a)



(b)

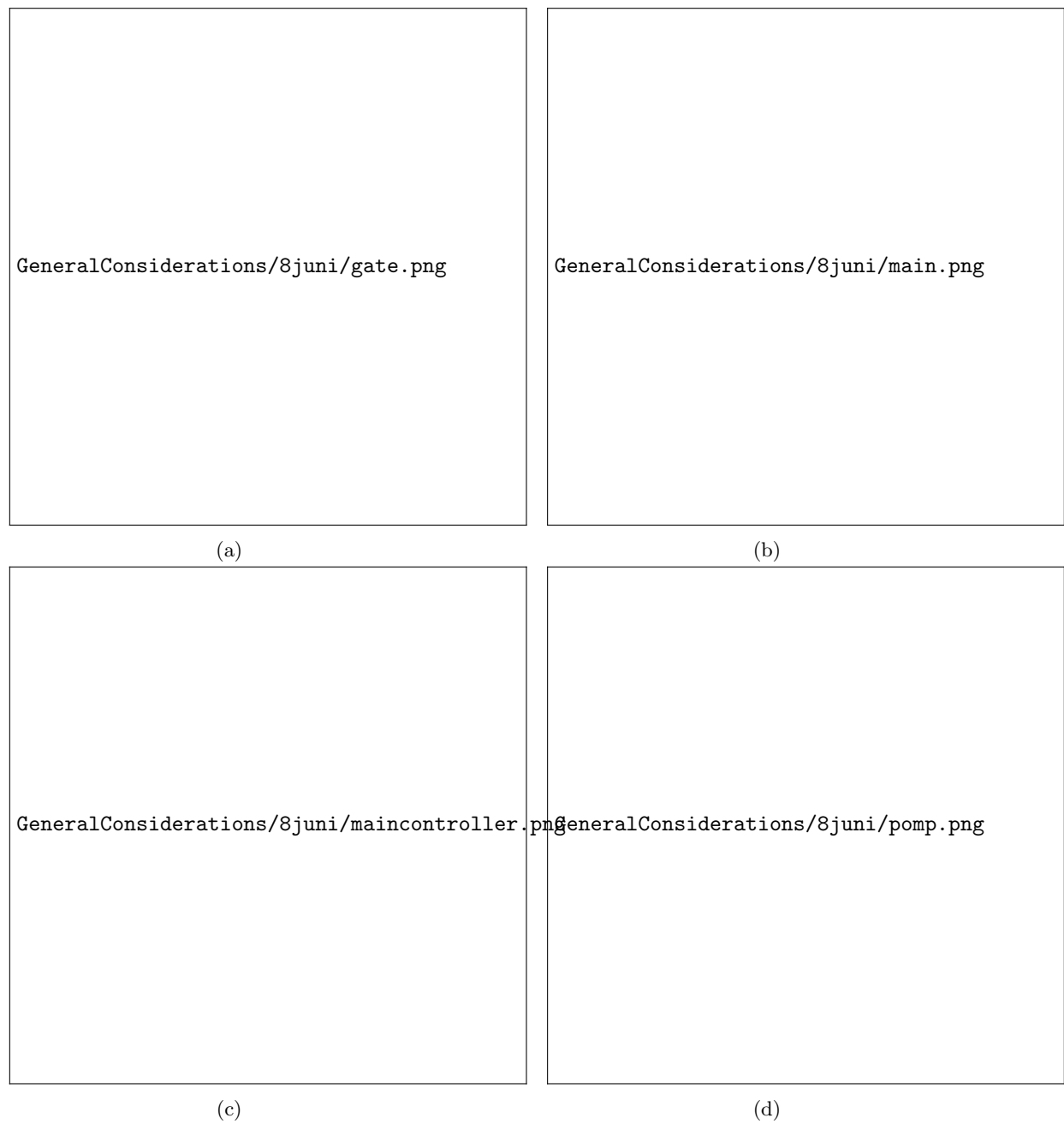


(c)

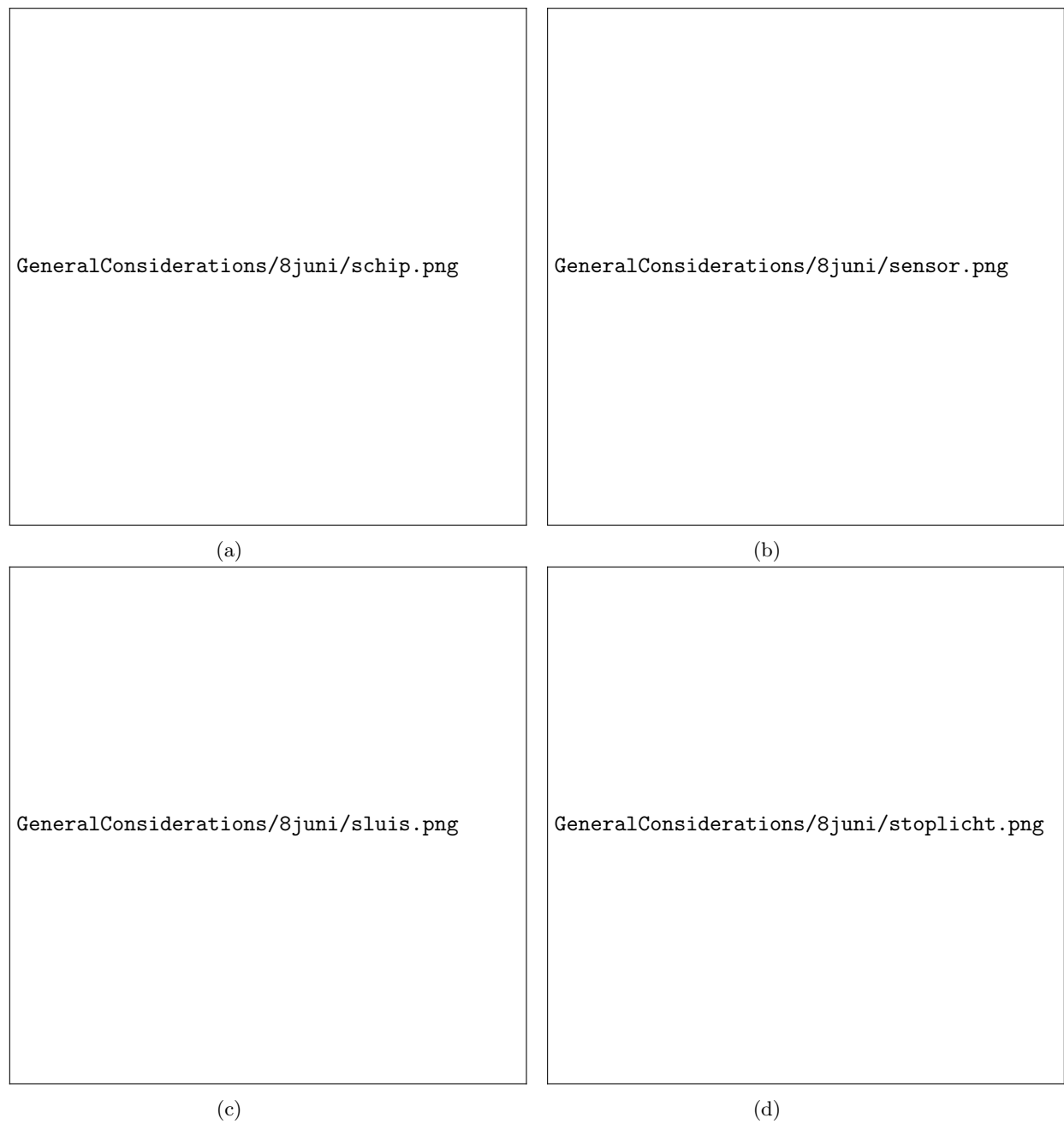


(d)

Figuur 18: Sluismodel 8 juni

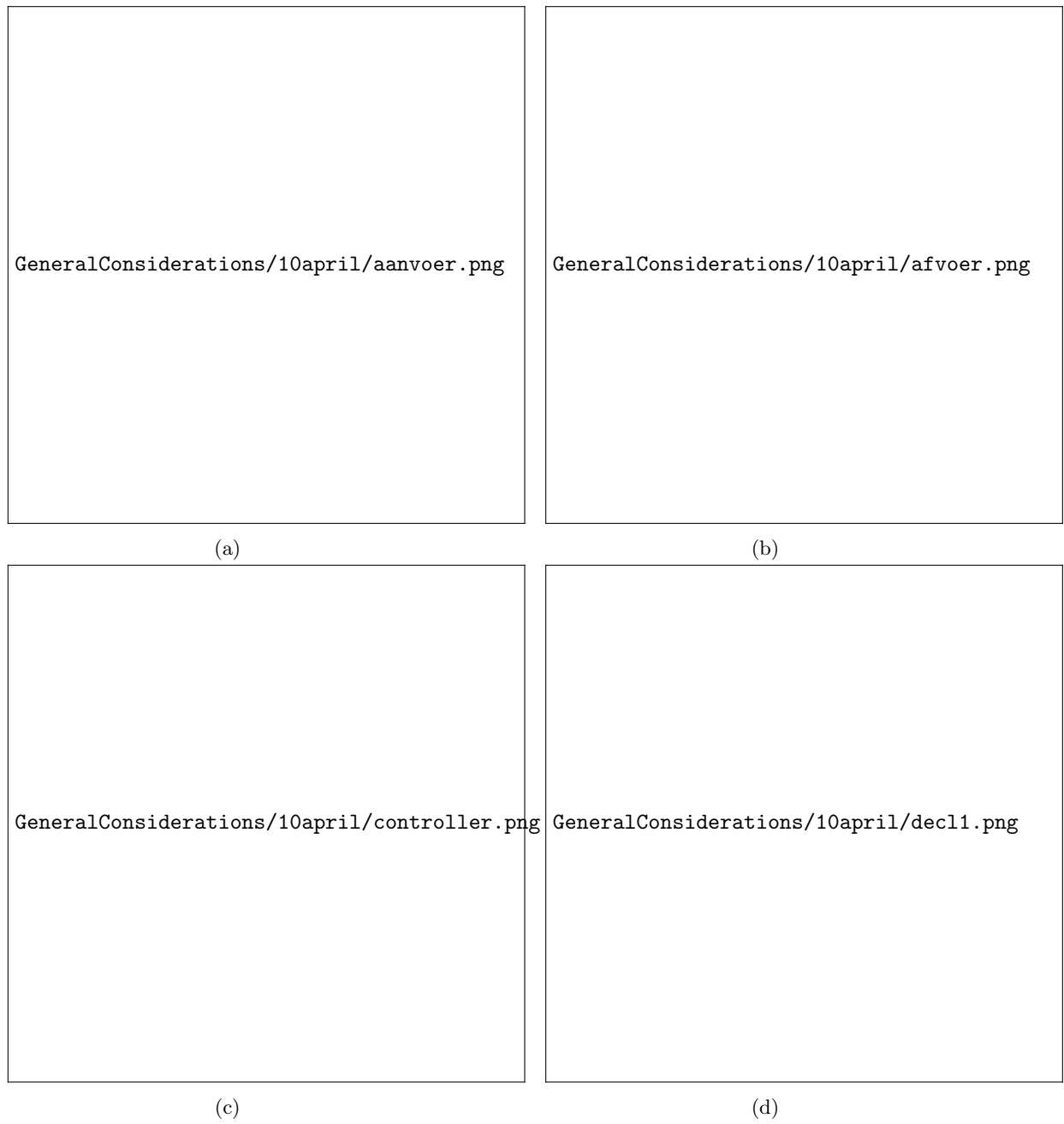


Figuur 19: Sluismodel 8 juni

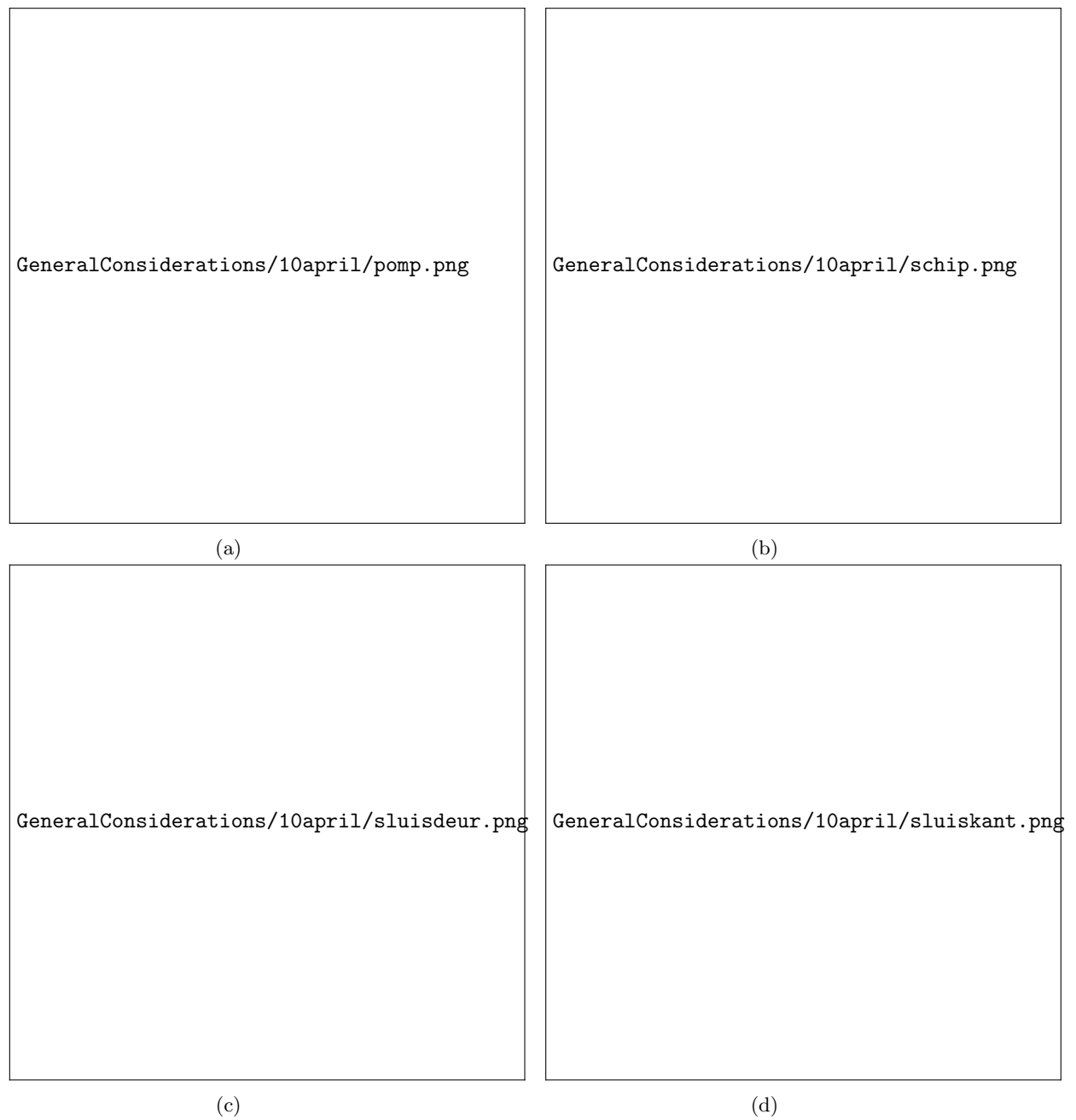


Figuur 20: Sluismodel 8 juni

B.6 Ontwerpen: 10 april



Figuur 21: Sluismodel 10 april

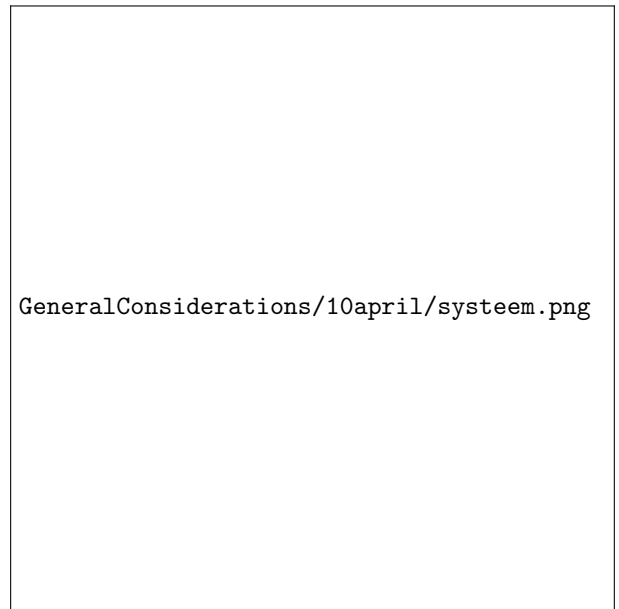


Figuur 22: Sluismodel 10 april



GeneralConsiderations/10april/stoplicht.png

(a)



GeneralConsiderations/10april/systeem.png

(b)

B.7 Ontwerpen: 12 juni



(a)



(b)

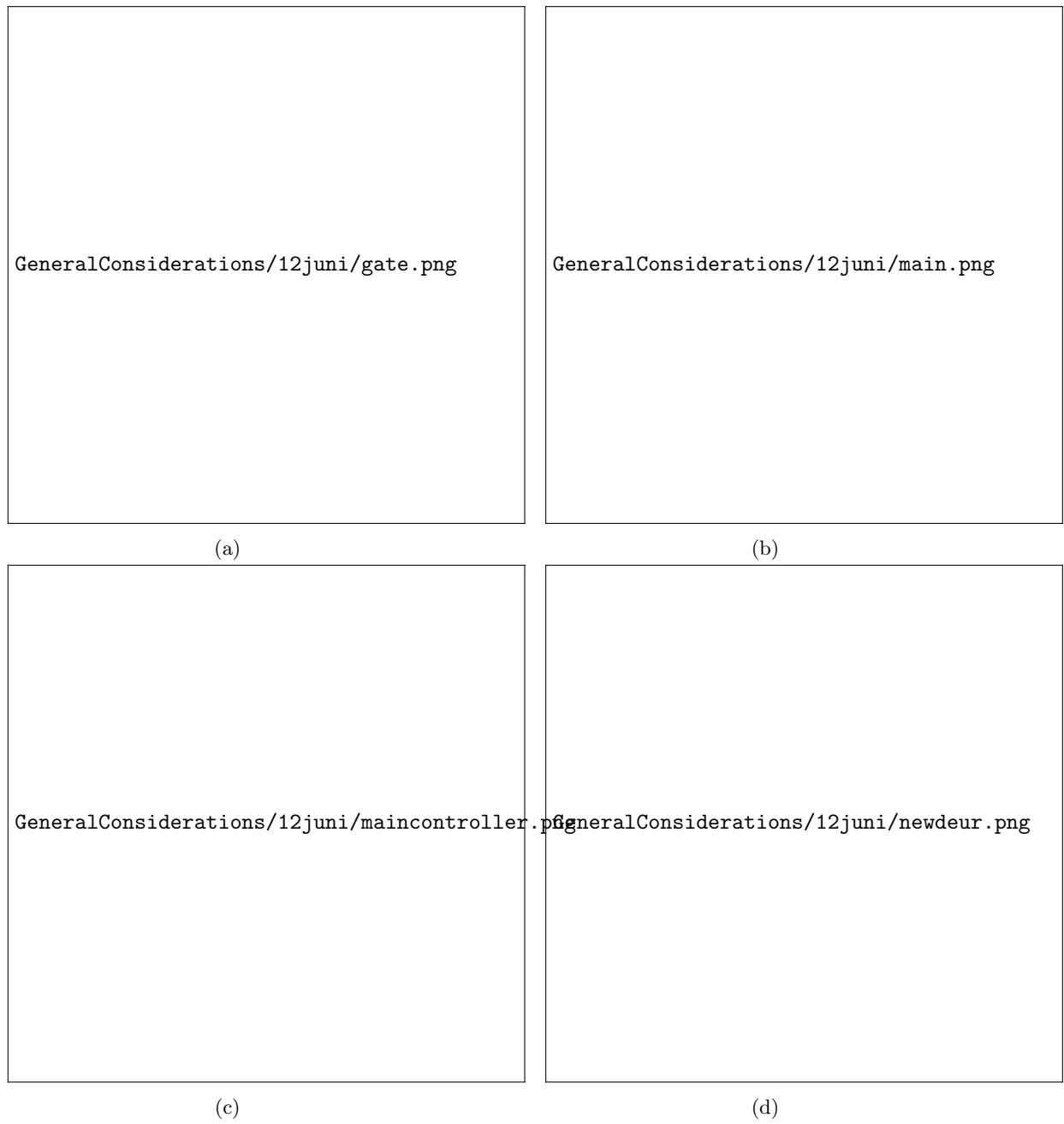


(c)

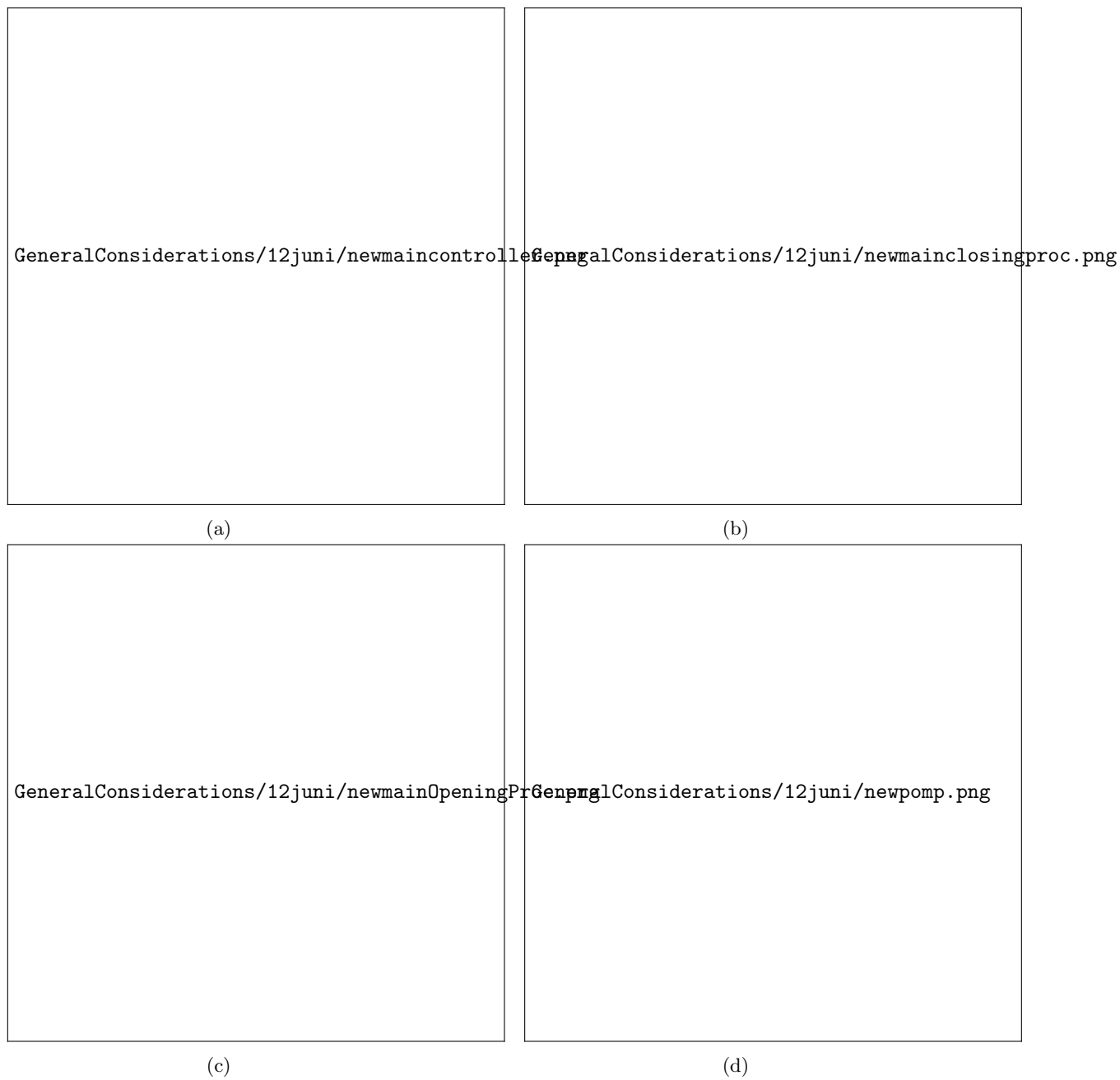


(d)

Figuur 24: Sluismodel 12 juni



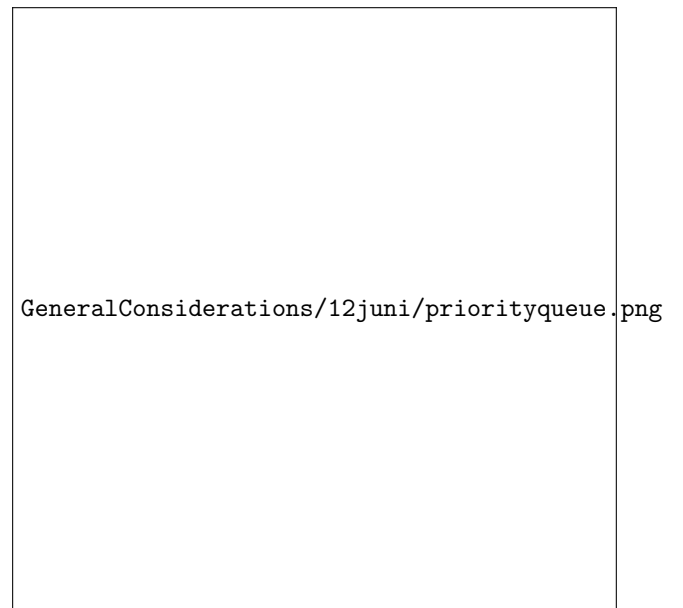
Figuur 25: Sluismodel 12 juni



Figuur 26: Sluismodel 12 juni



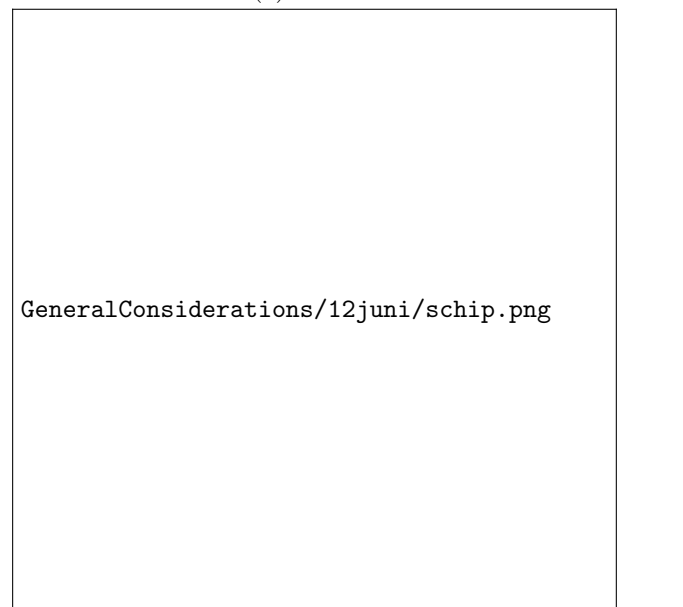
(a)



(b)

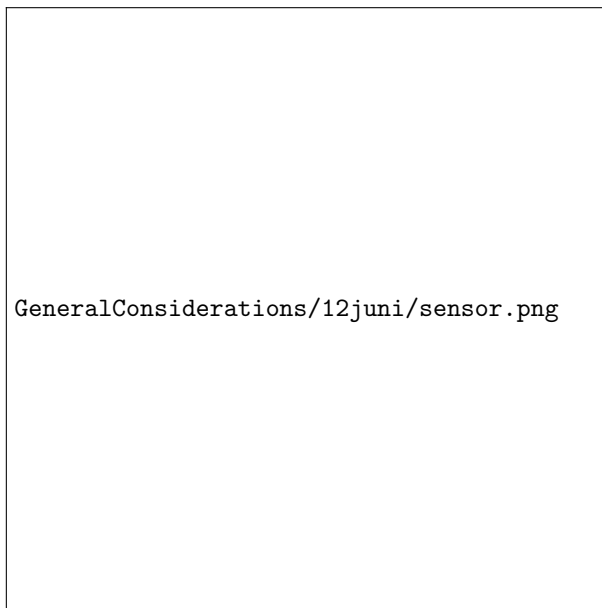


(c)

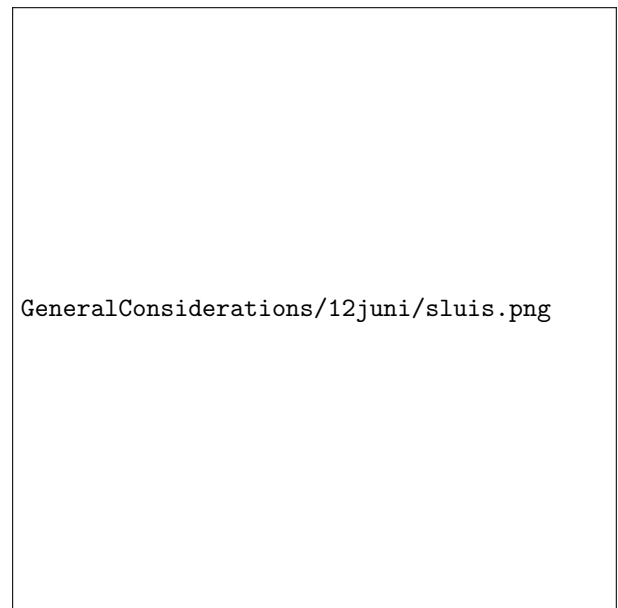


(d)

Figuur 27: Sluismodel 12 juni



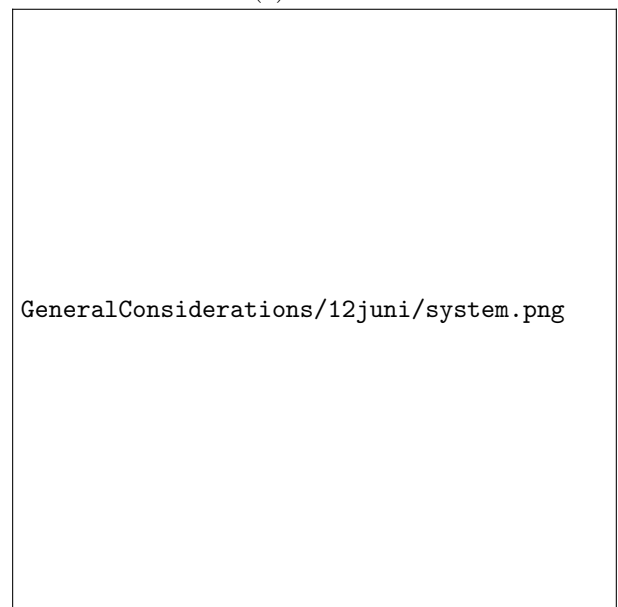
(a)



(b)



(c)



(d)

Figuur 28: Sluismodel 12 juni



(a)

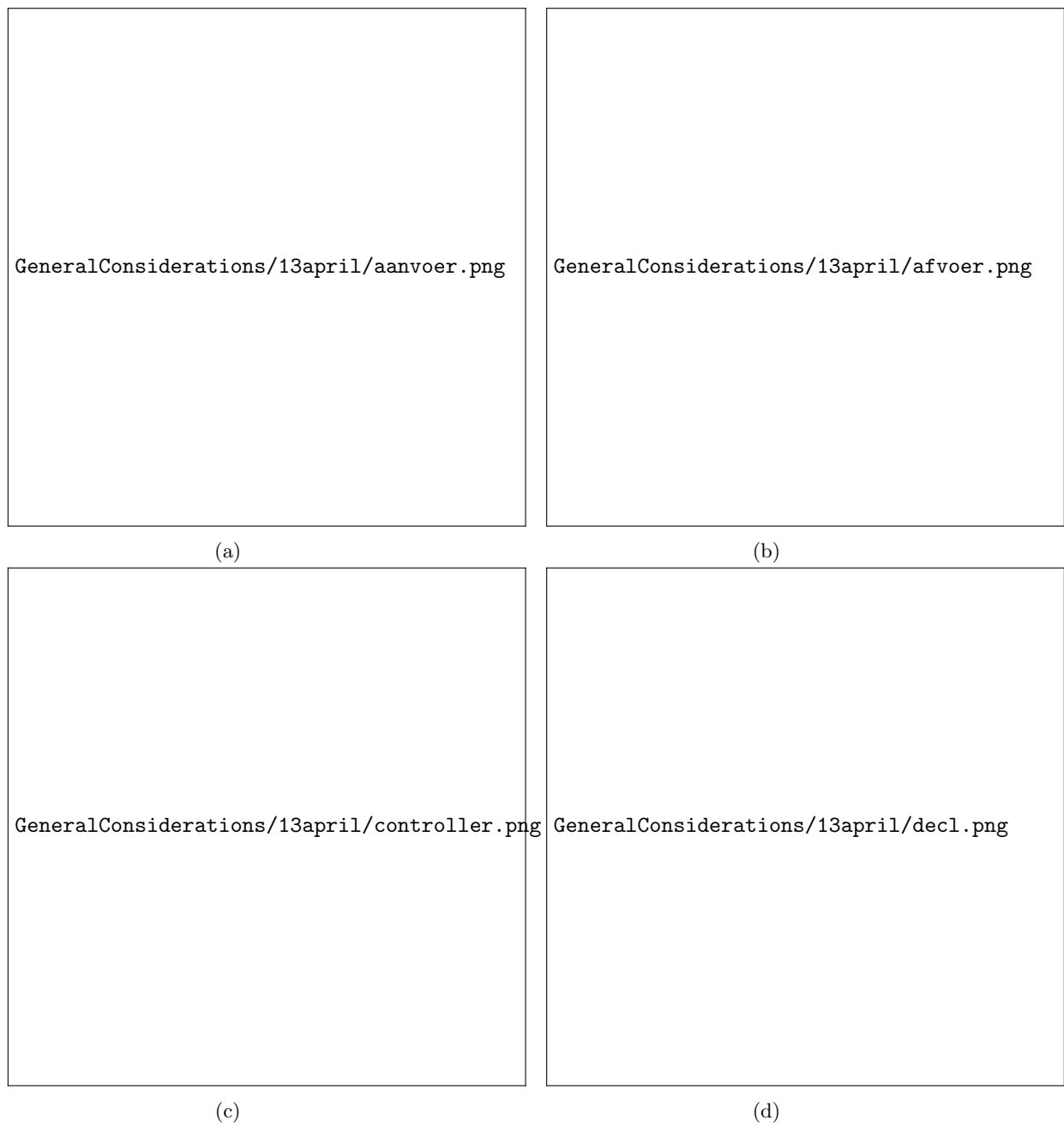


(b)

B.8 Ontwerpen: 12 september



B.9 Ontwerpen: 13 april



Figuur 30: Sluismodel 13 april



(a)



(b)



(c)

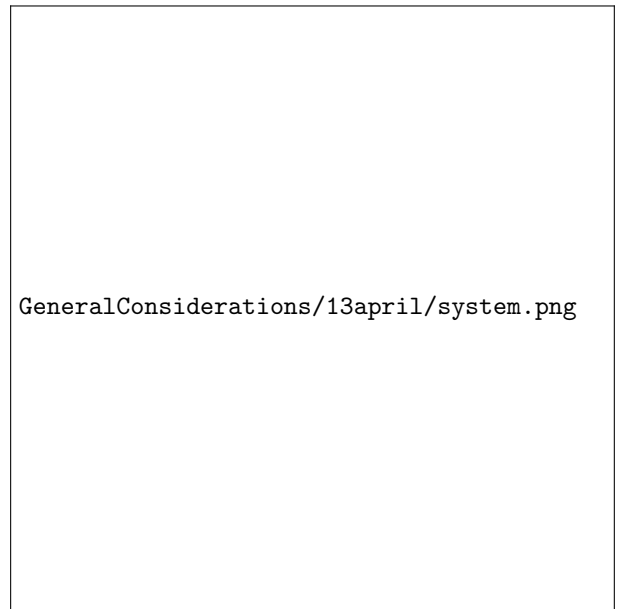


(d)

Figuur 31: Sluismodel 13 april

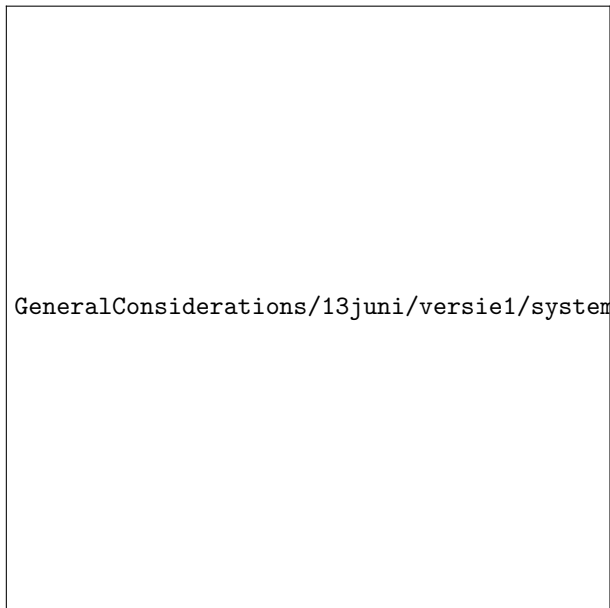


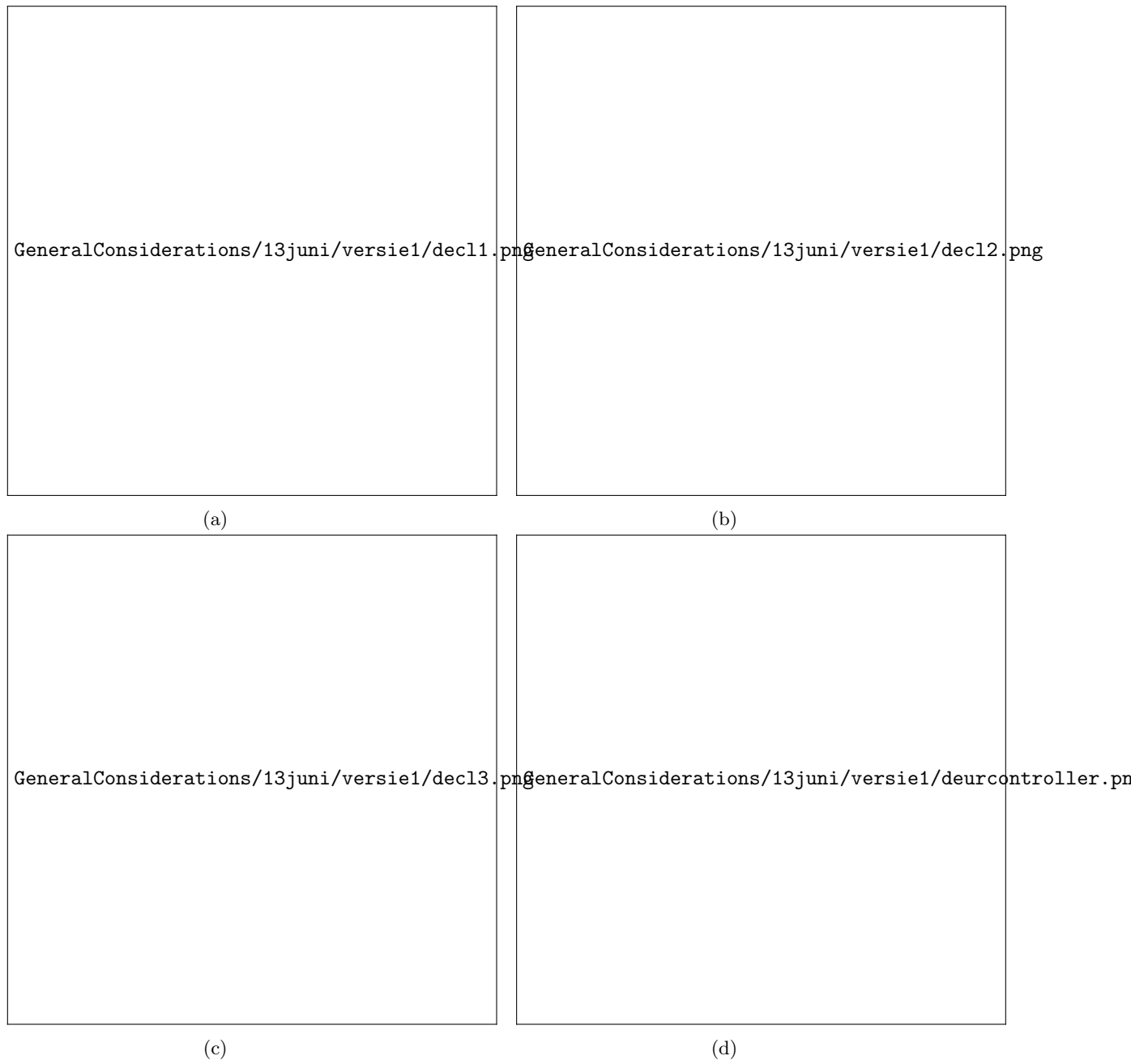
(a)



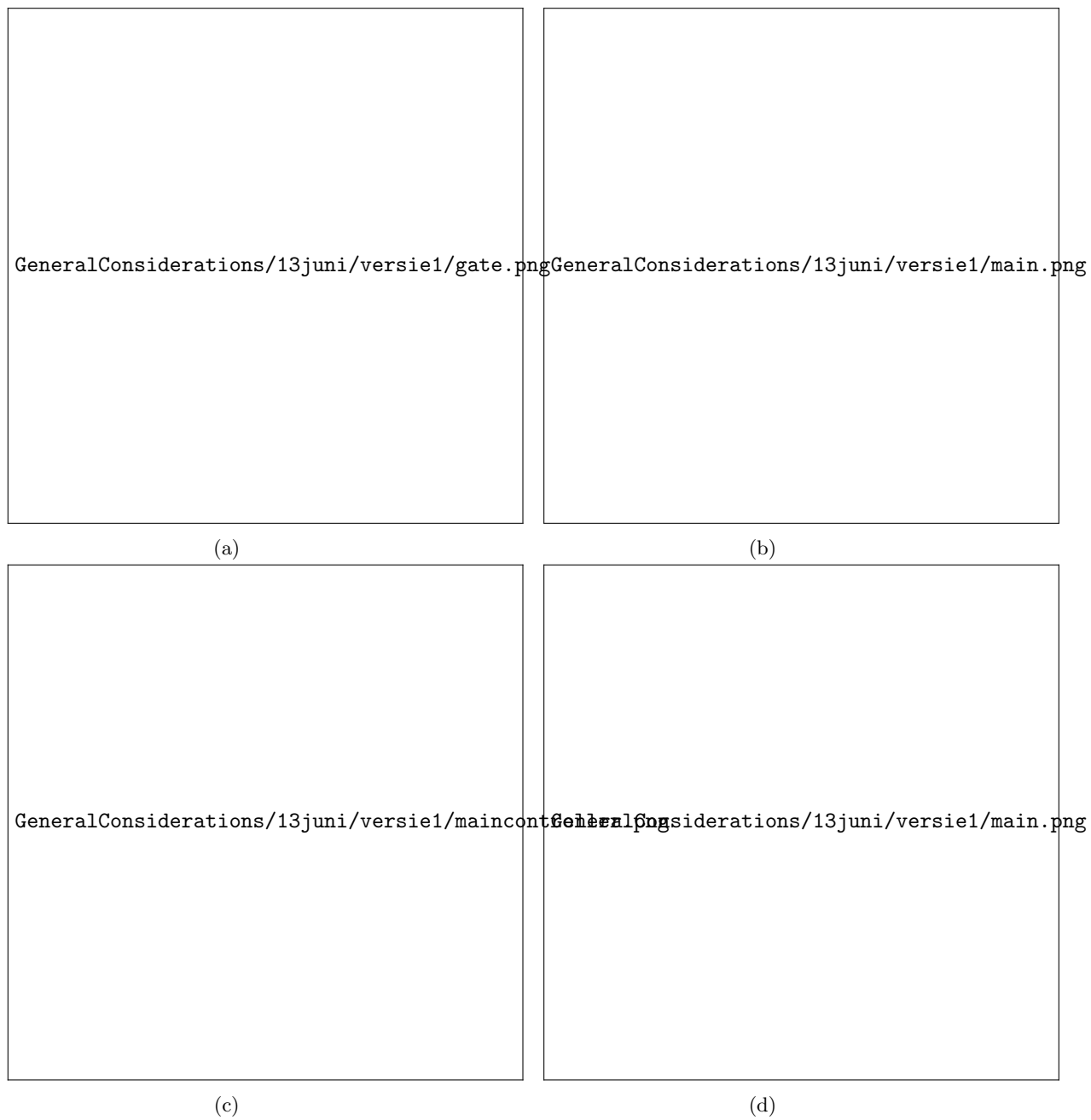
(b)

B.10 Ontwerpen: 13 juni

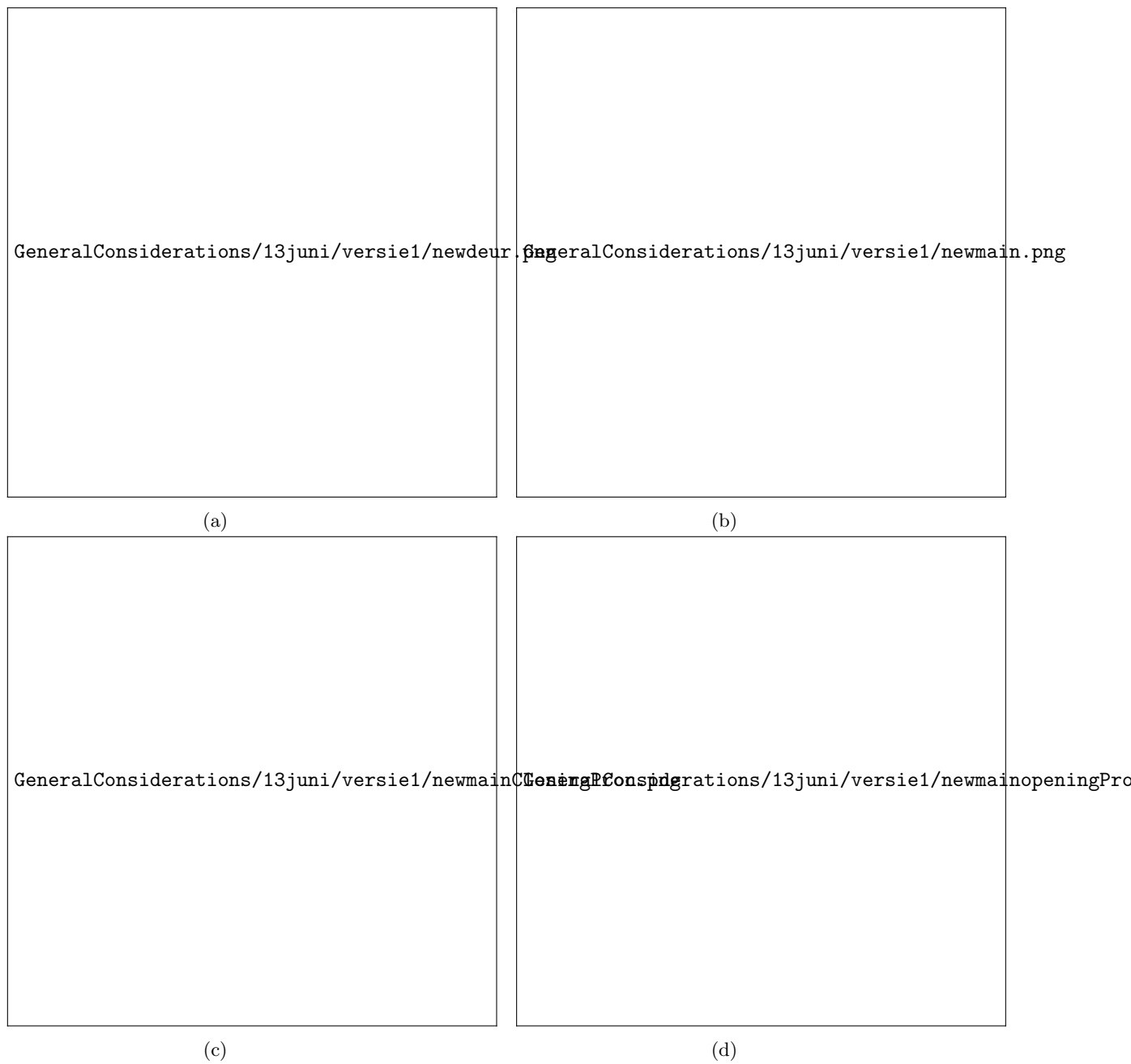




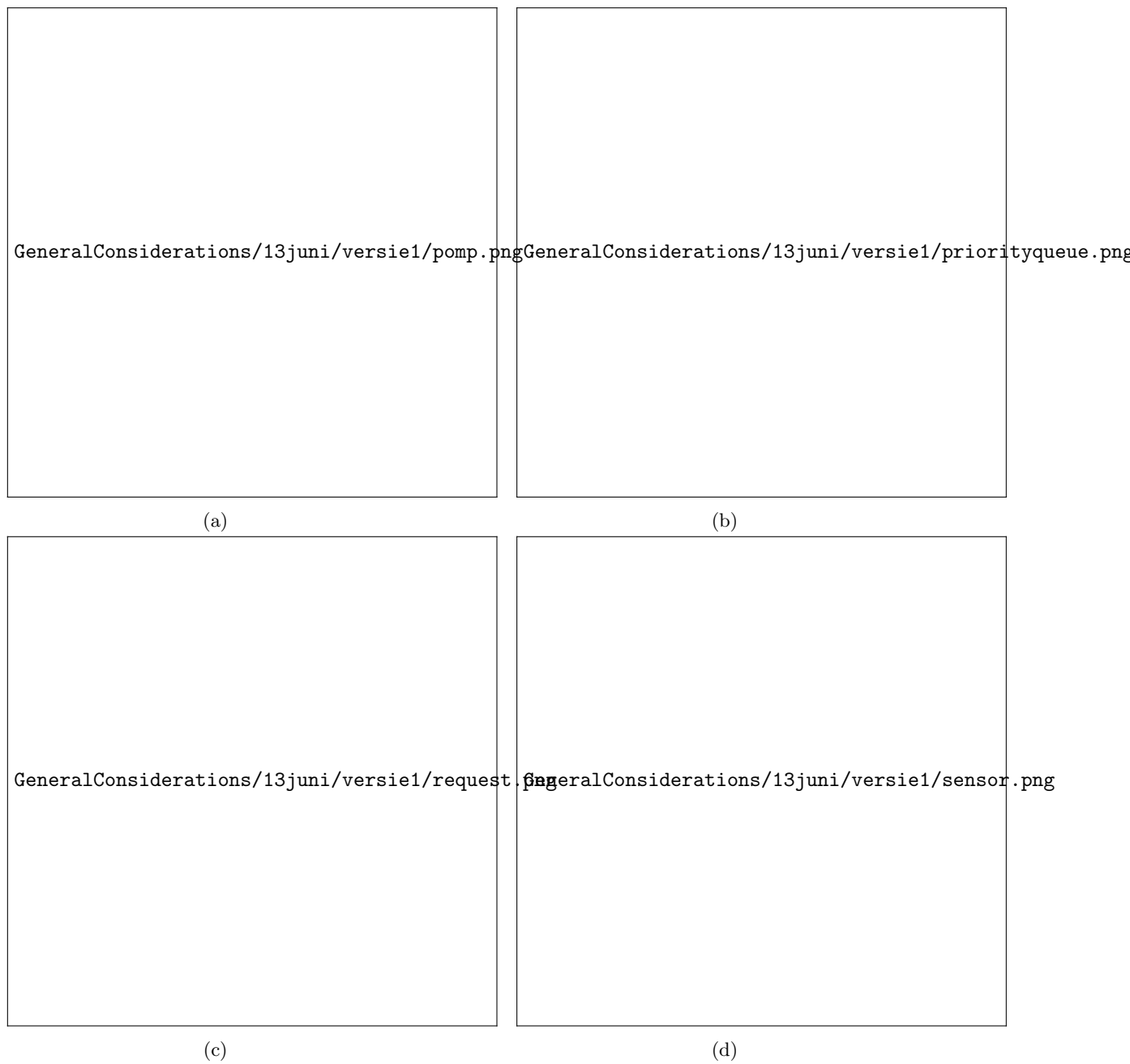
Figuur 33: Sluismodel 13 juni versie 1



Figuur 34: Sluismodel 13 juni versie 1



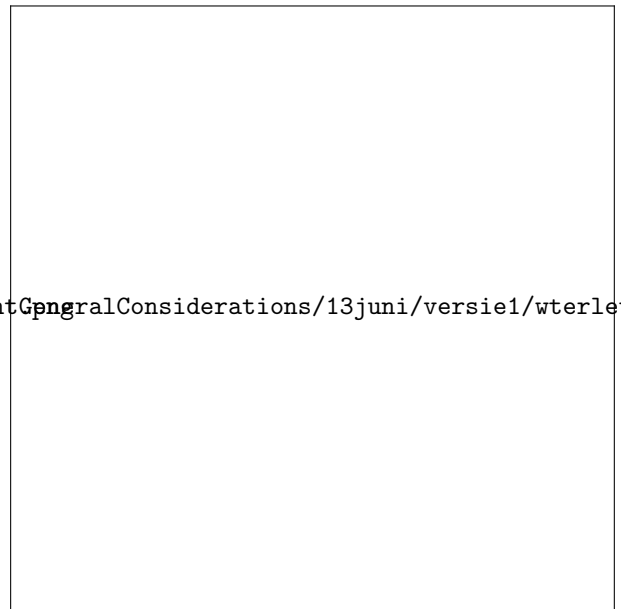
Figuur 35: Sluismodel 13 juni versie 1



Figuur 36: Sluismodel 13 juni versie 1



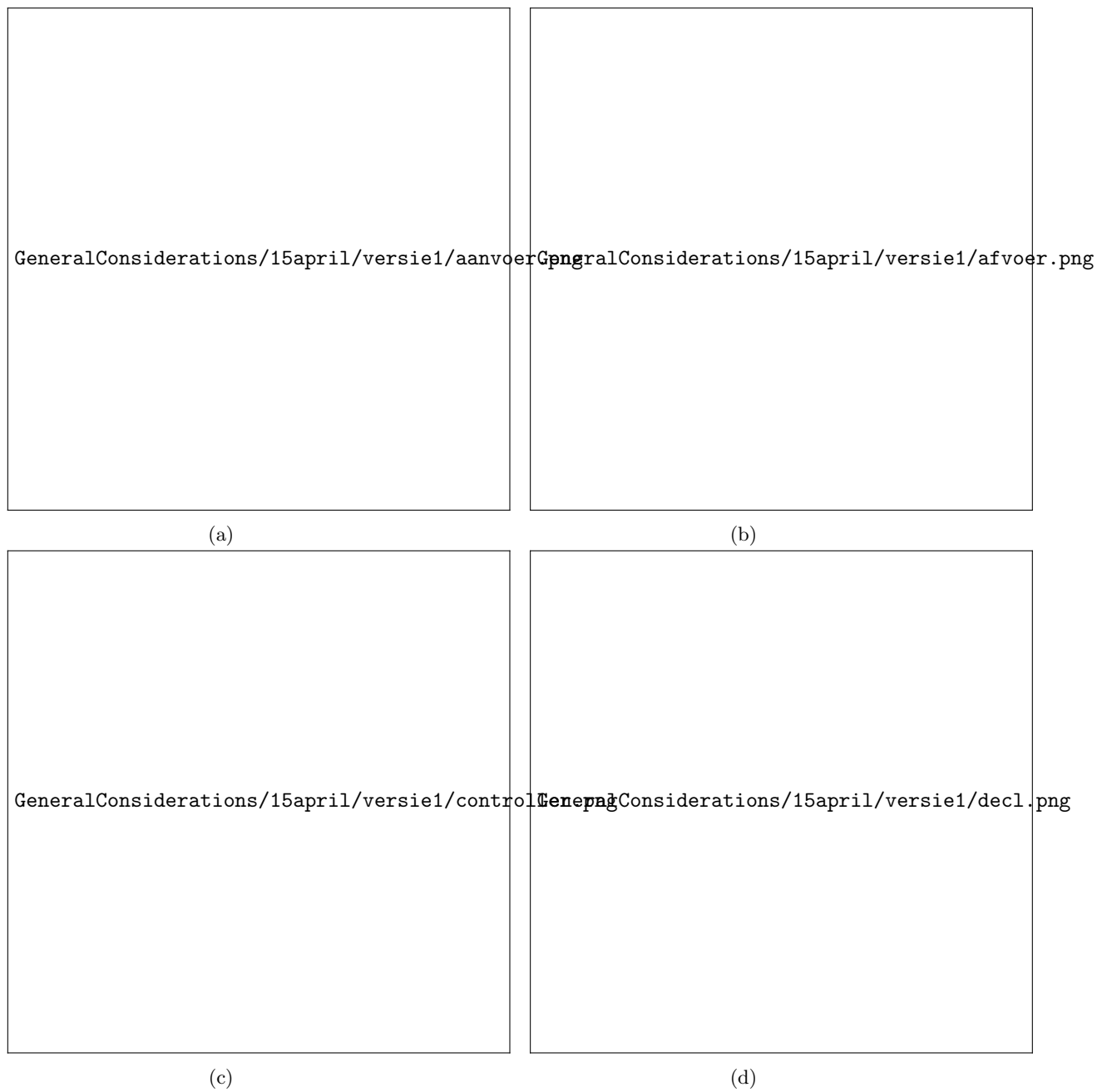
(a)



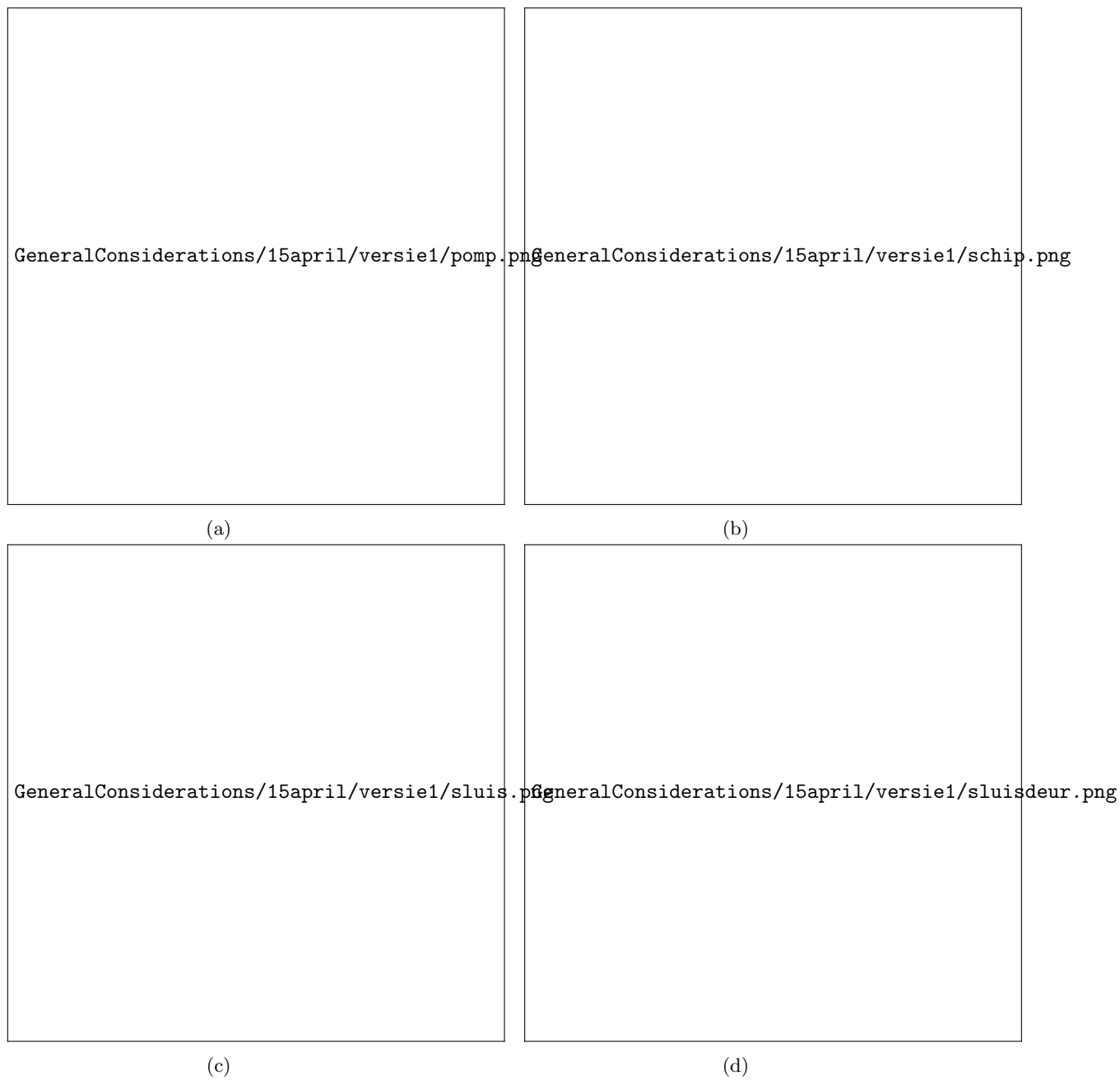
(b)

B.11 Ontwerpen: 15 april





Figuur 38: Sluismodel 15 april versie 1



Figuur 39: Sluismodel 15 april versie 1

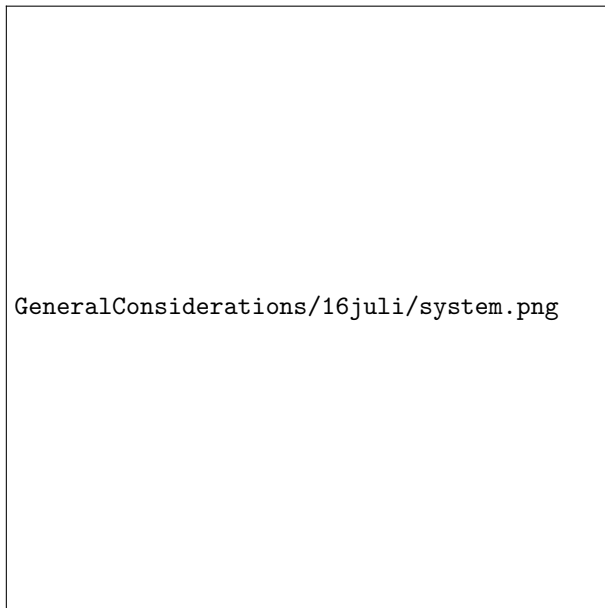


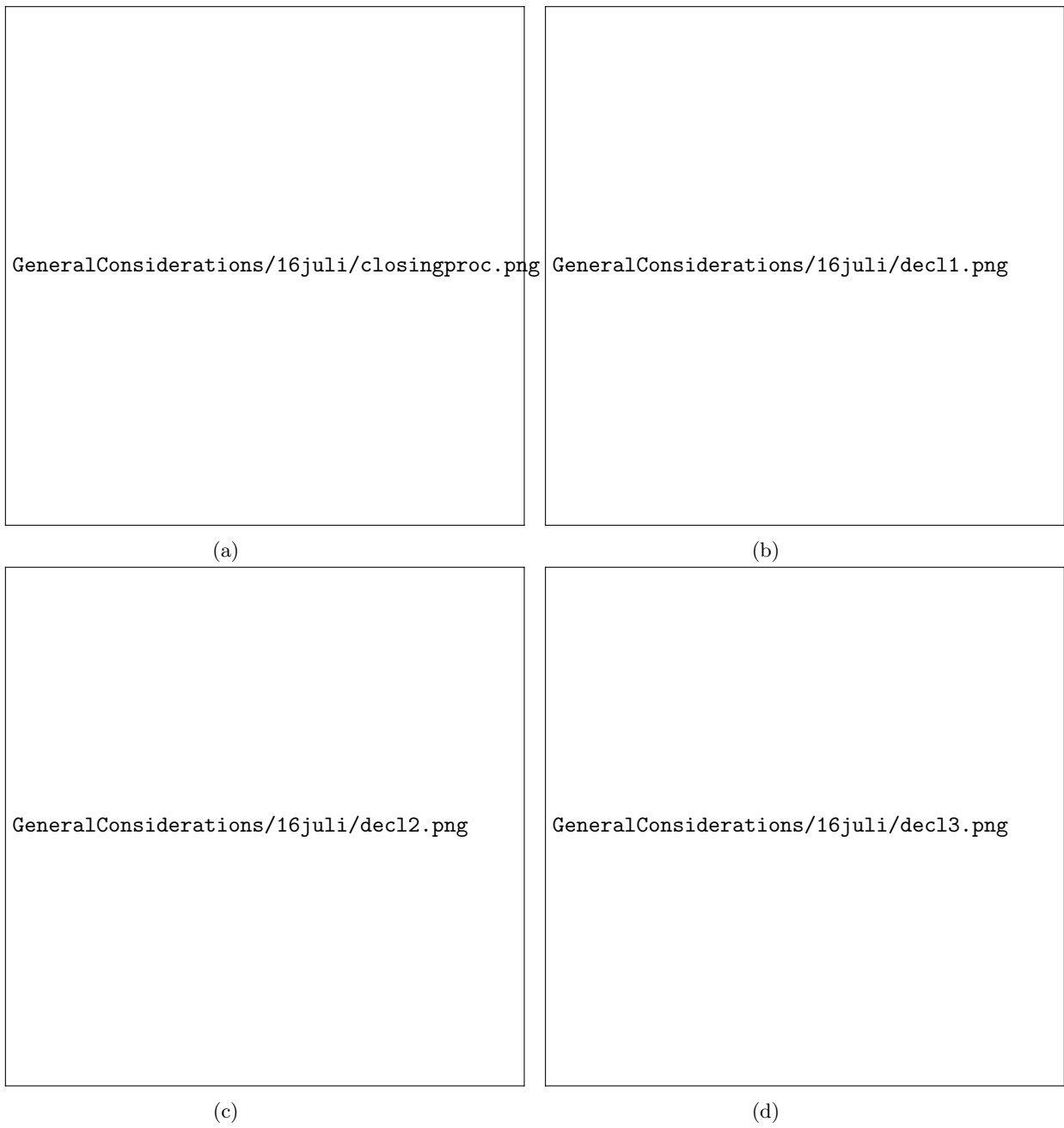
(a)



(b)

B.12 Ontwerpen:16 juli





Figuur 41: Sluismodel 16 juli



(a)



(b)



(c)

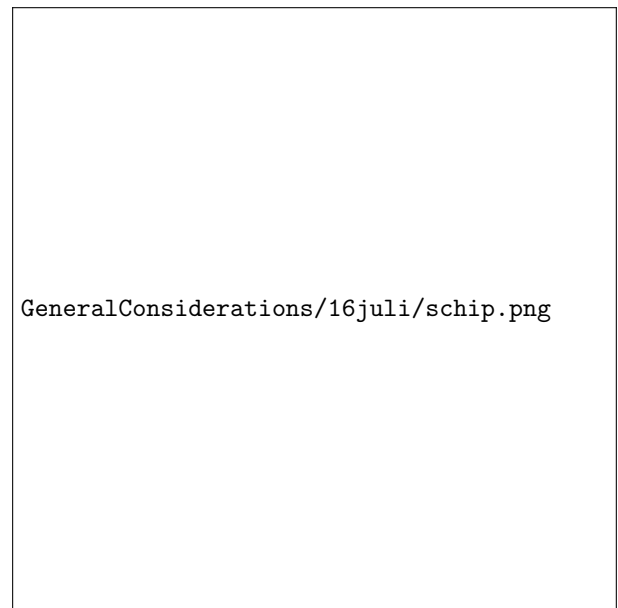


(d)

Figuur 42: Sluismodel 16 juli



(a)



(b)



(c)

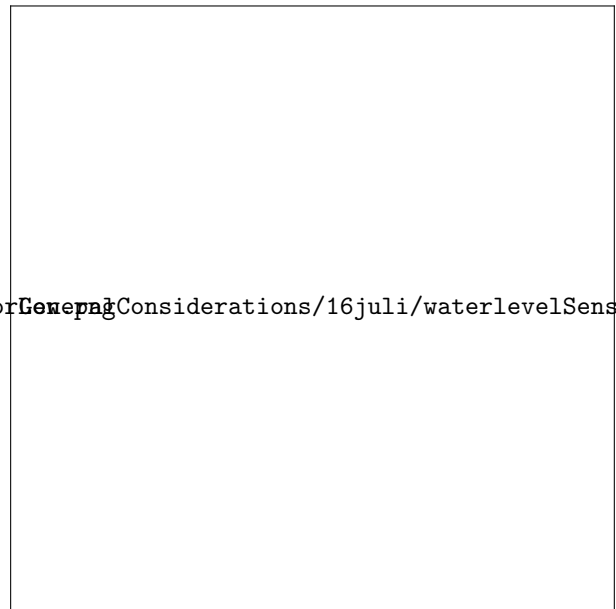


(d)

Figuur 43: Sluismodel 16 juli



(a)



(b)

B.13 Ontwerpen: 17 april



(a)



(b)



(c)

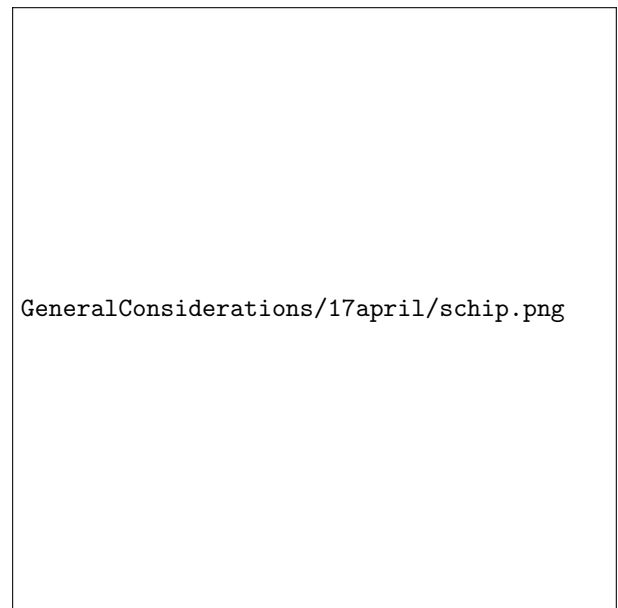


(d)

Figuur 45: Sluismodel 17 april



(a)



(b)



(c)



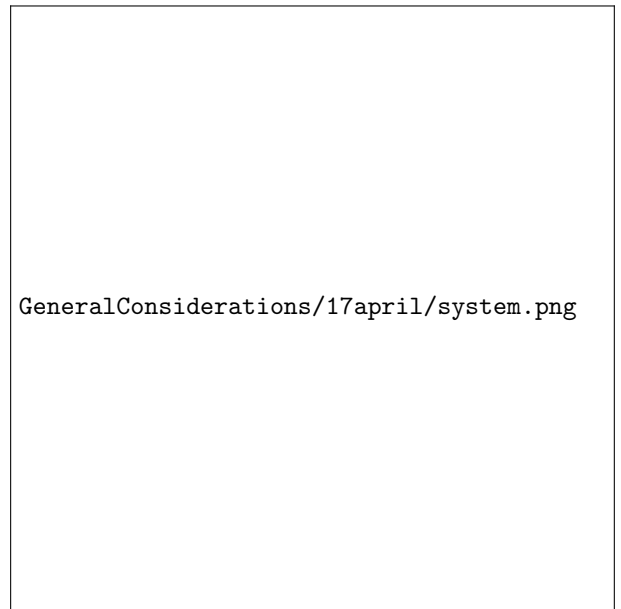
(d)

Figuur 46: Sluismodel 17 april



GeneralConsiderations/17april/stoplight.png

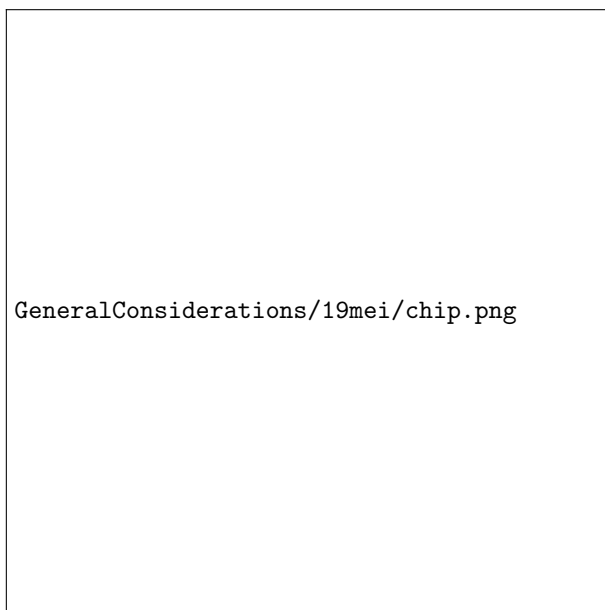
(a)



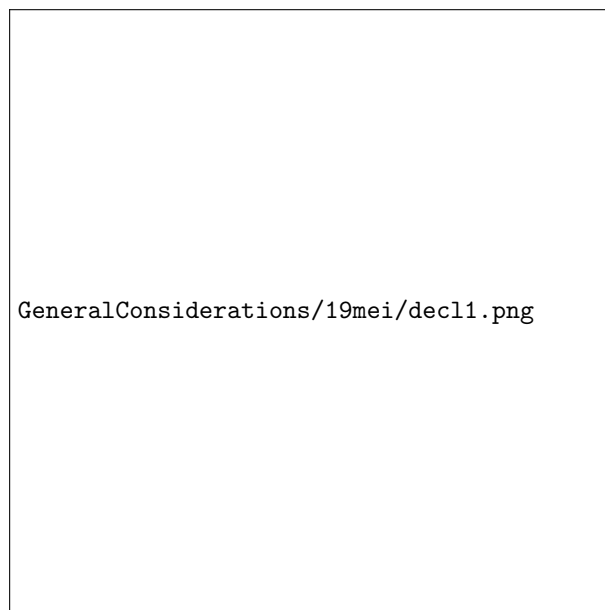
GeneralConsiderations/17april/system.png

(b)

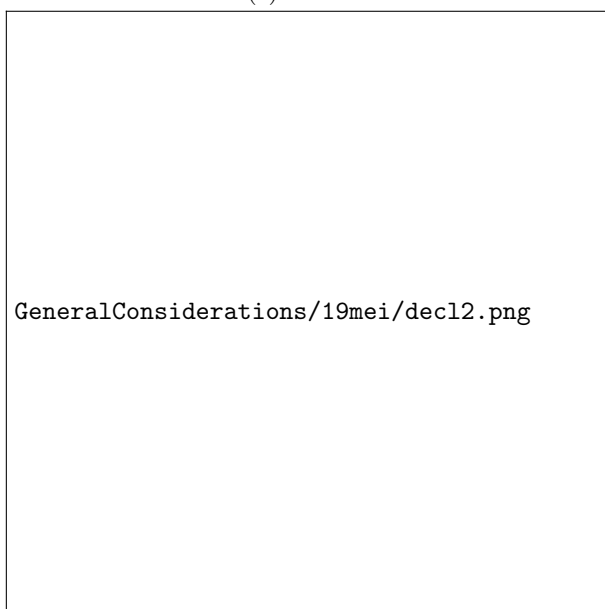
B.14 Ontwerpen: 1 mei



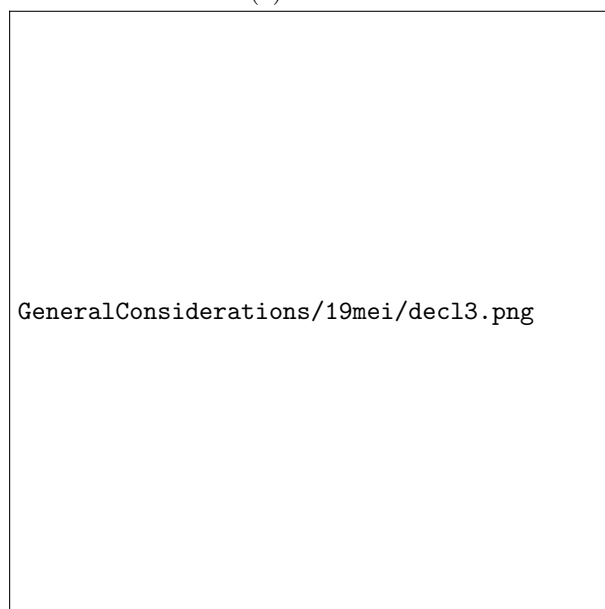
(a)



(b)



(c)

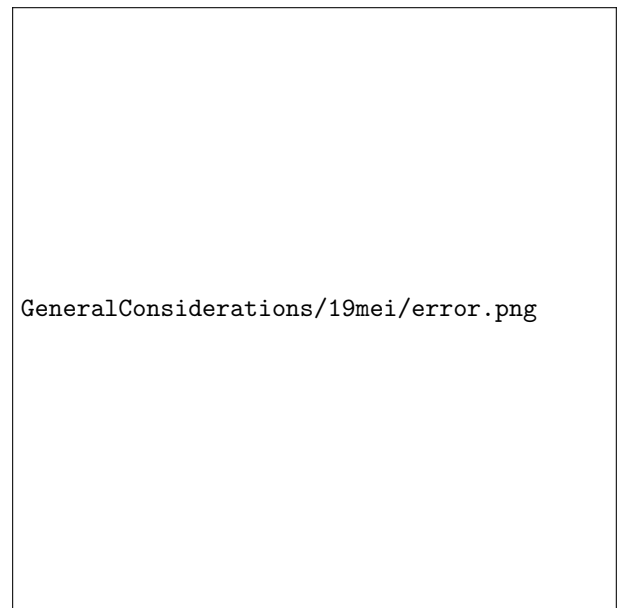


(d)

Figuur 48: Sluismodel 19 mei



(a)



(b)

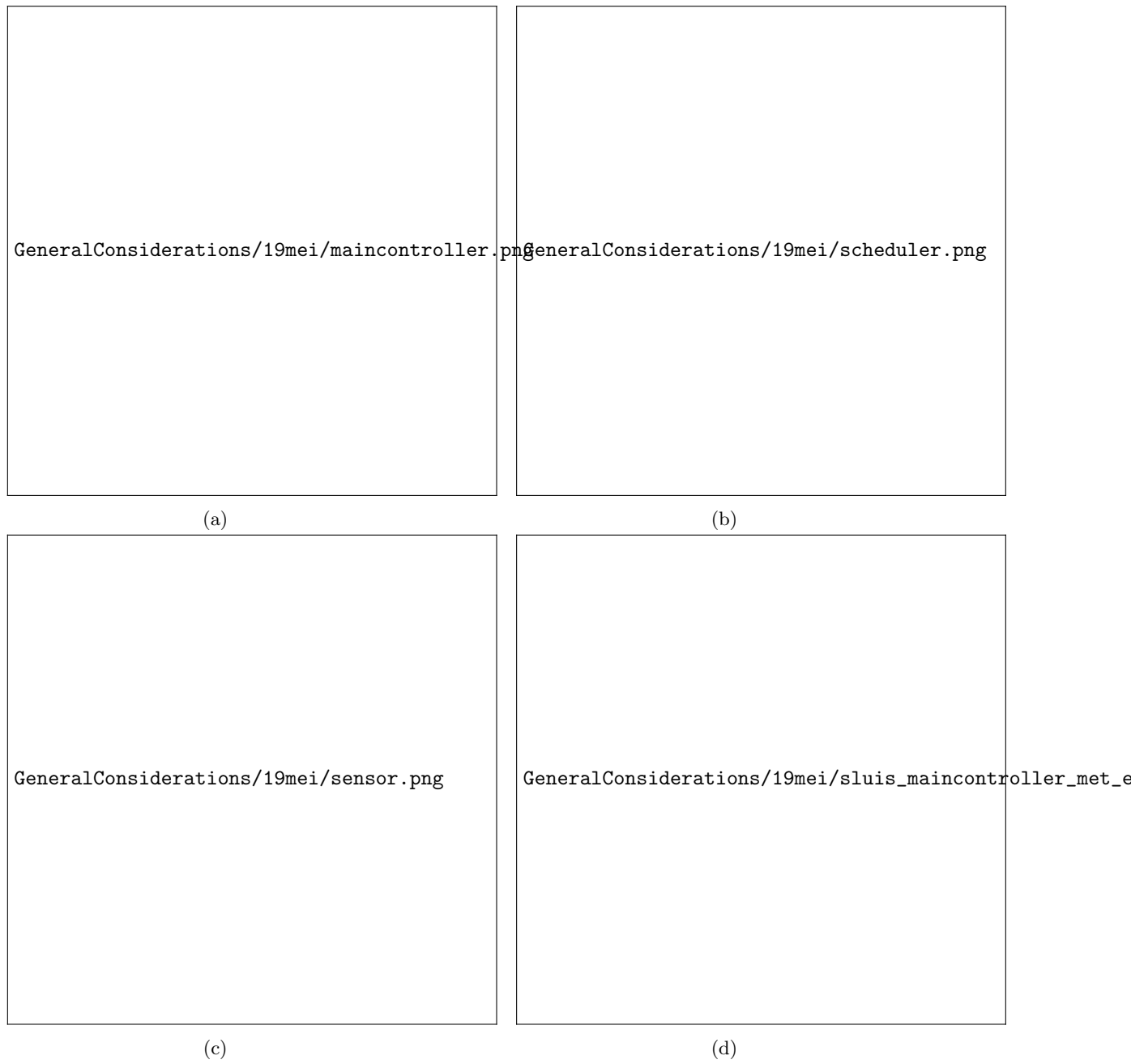


(c)



(d)

Figuur 49: Sluismodel 19 mei



Figuur 50: Sluismodel 19 mei



(a)



(b)



(c)



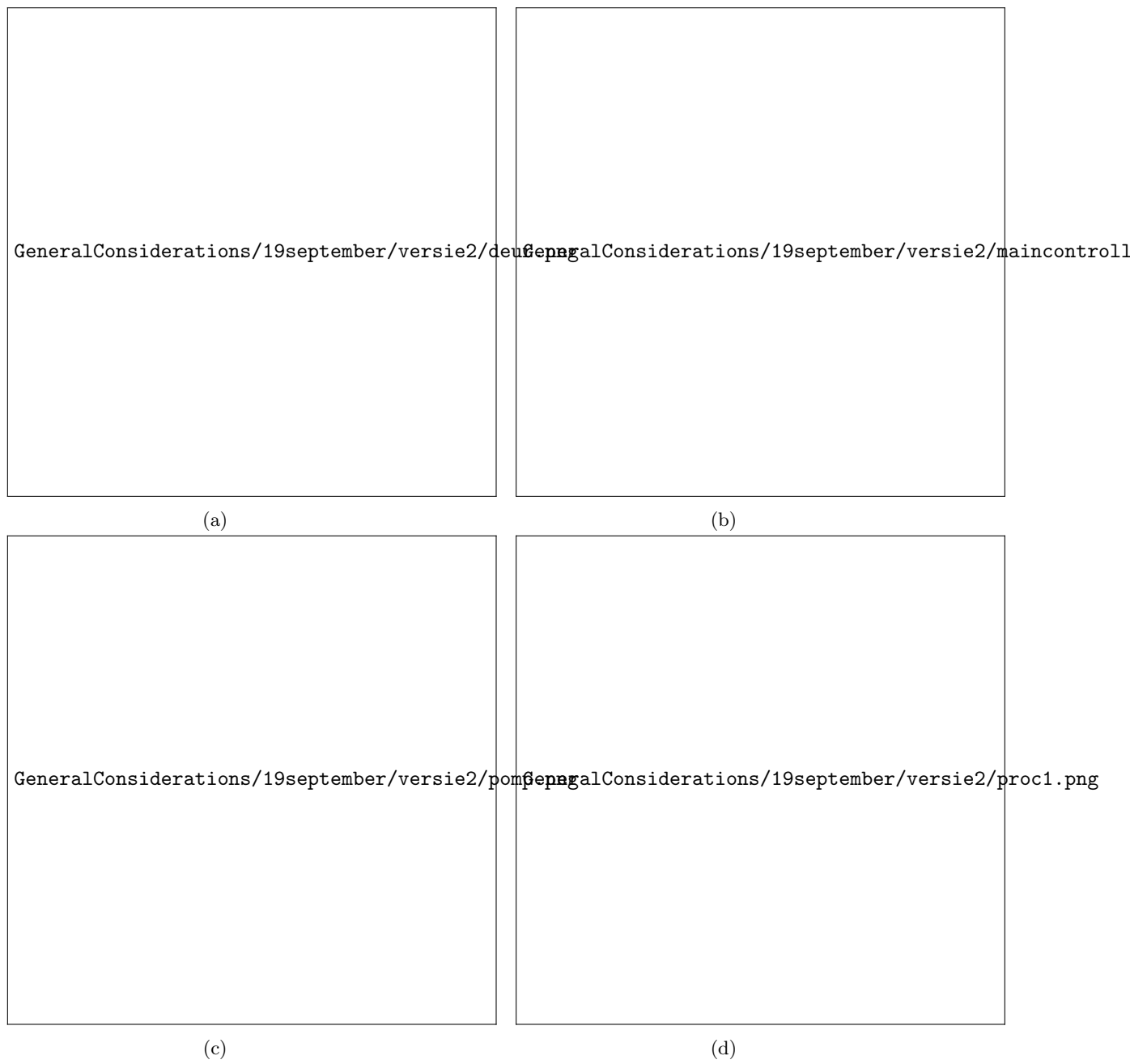
(d)

Figuur 51: Sluismodel 19 mei

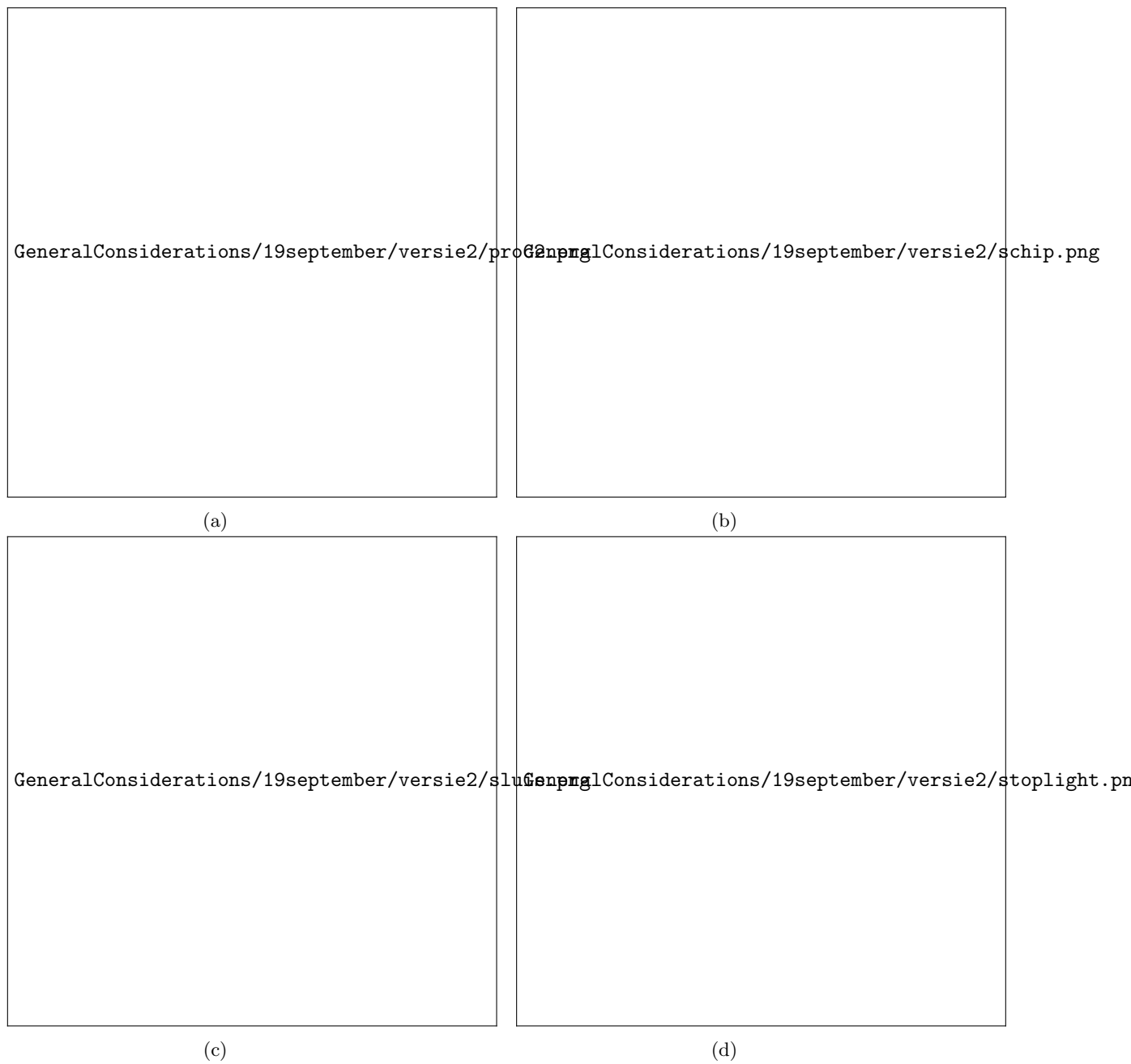
B.15 Ontwerpen: 19 september versie 2



GeneralConsiderations/19september/versie2/system.png

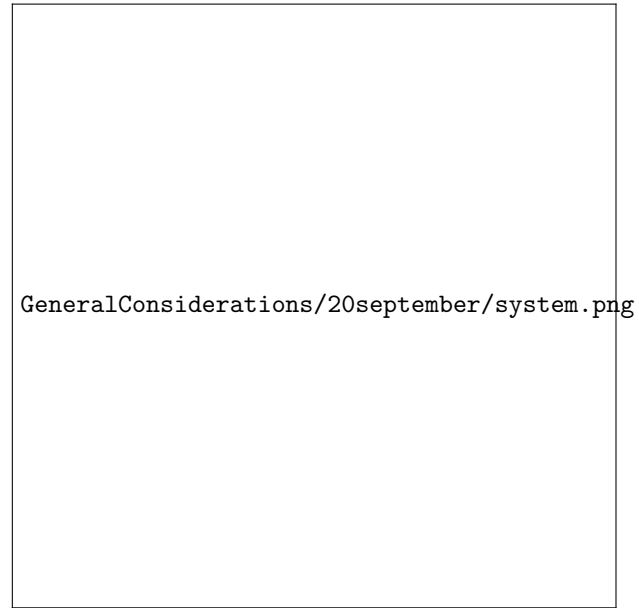


Figuur 52: Sluismodel 19 september



Figuur 53: Sluismodel 19 september

B.16 Ontwerpen: 20 september





(a)



(b)

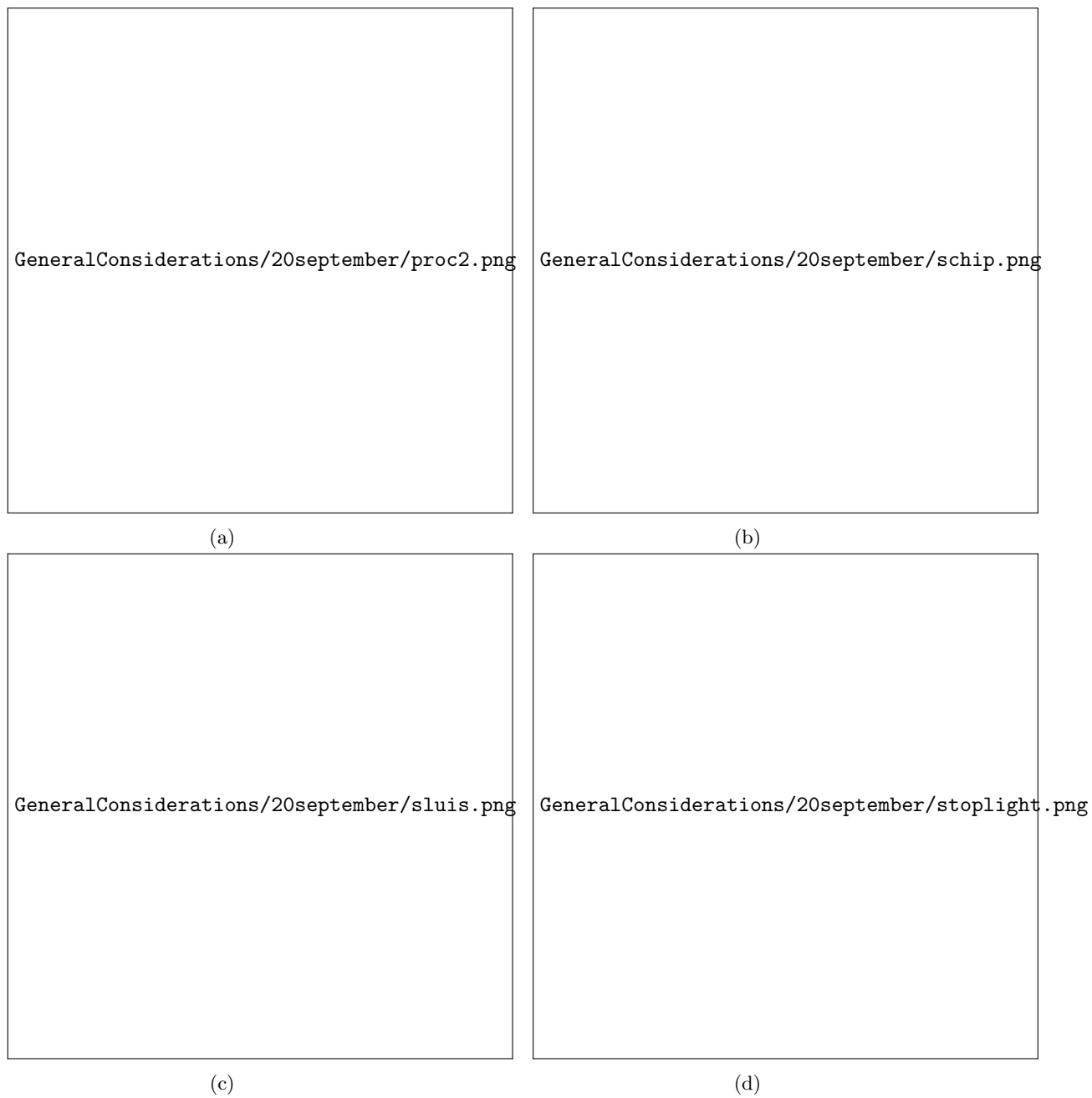


(c)



(d)

Figuur 54: Sluismodel 20 september



Figuur 55: Sluismodel 20 september

B.17 Ontwerpen: 22 mei



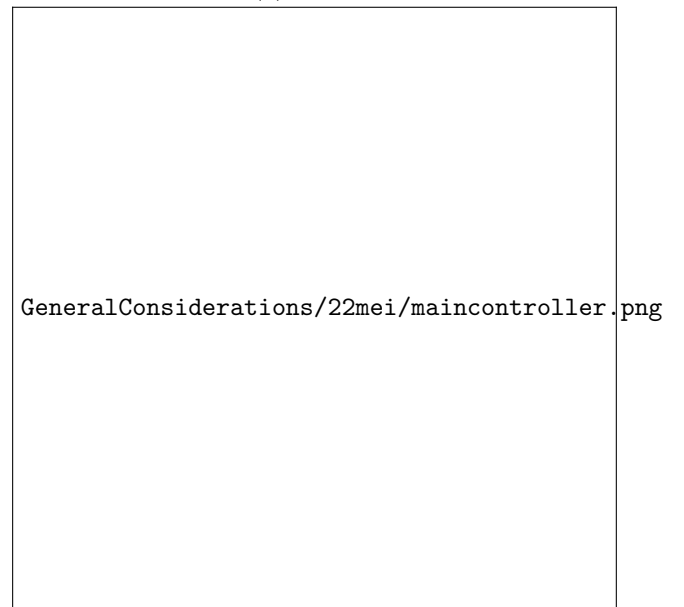
(a)



(b)



(c)

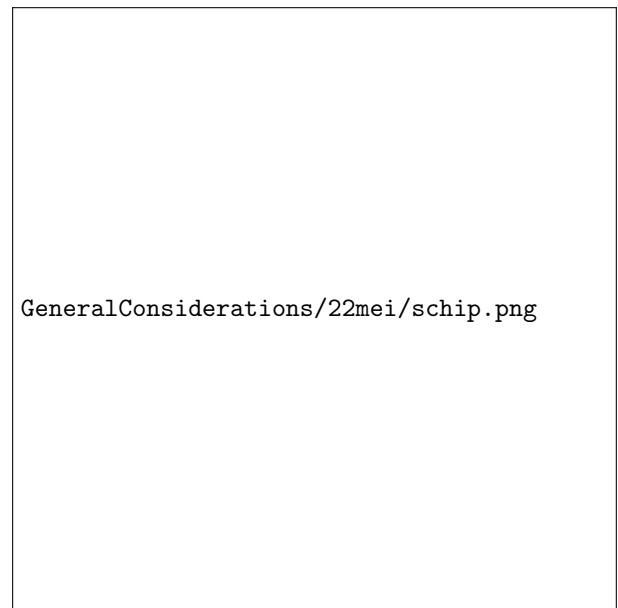


(d)

Figuur 56: Sluismodel 22 mei



(a)



(b)



(c)

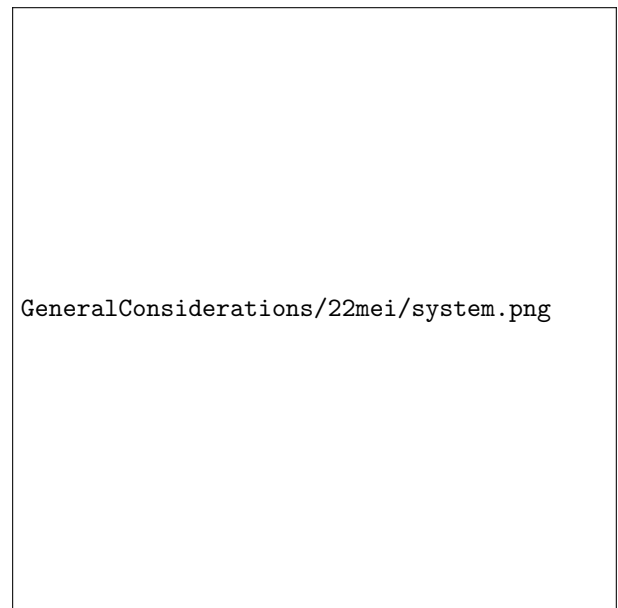


(d)

Figuur 57: Sluismodel 22 mei



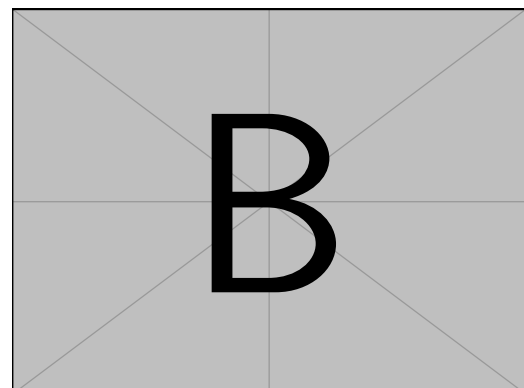
(a)



(b)



(c)



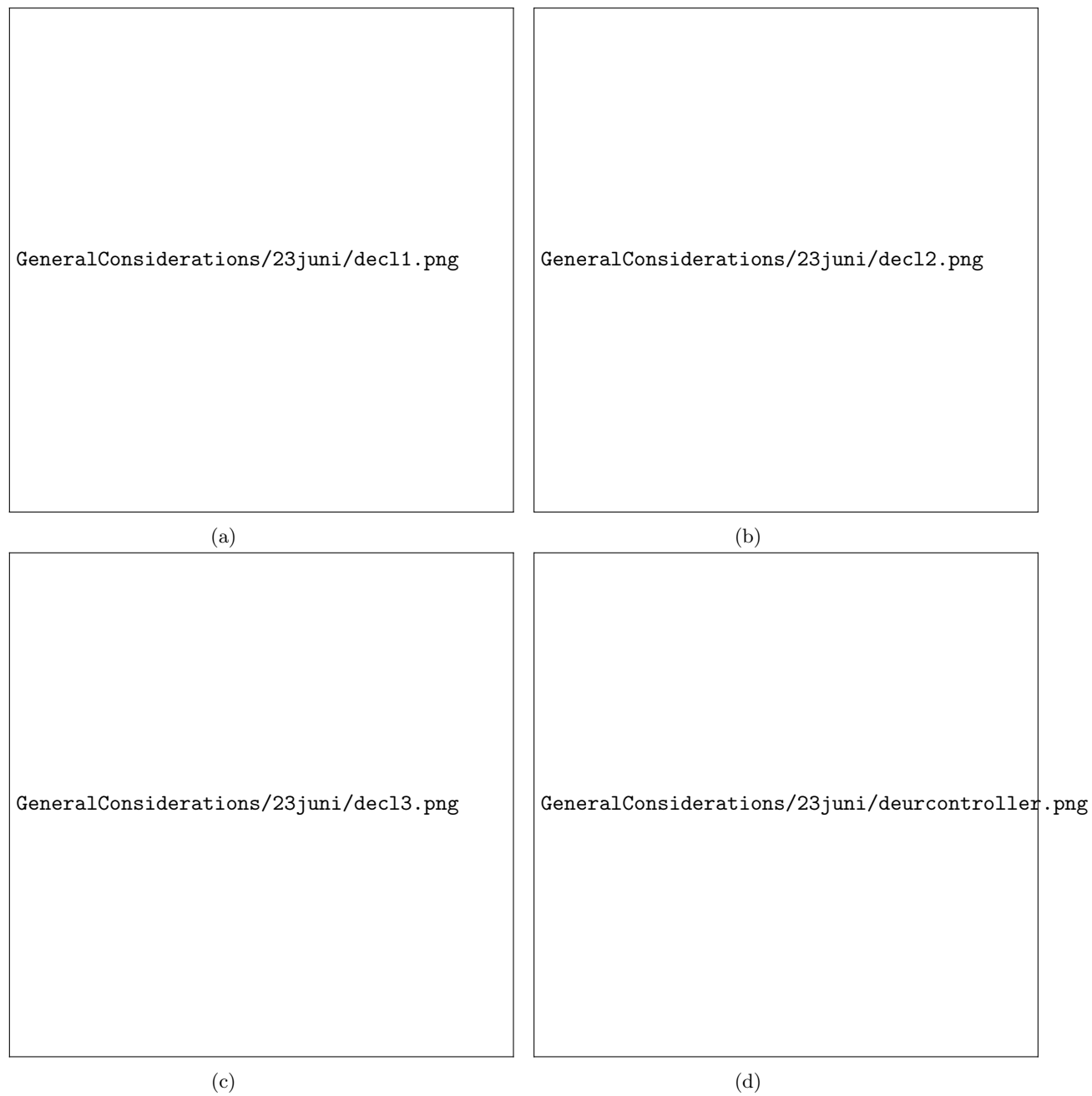
(d)

Figuur 58: Sluismodel 22 mei

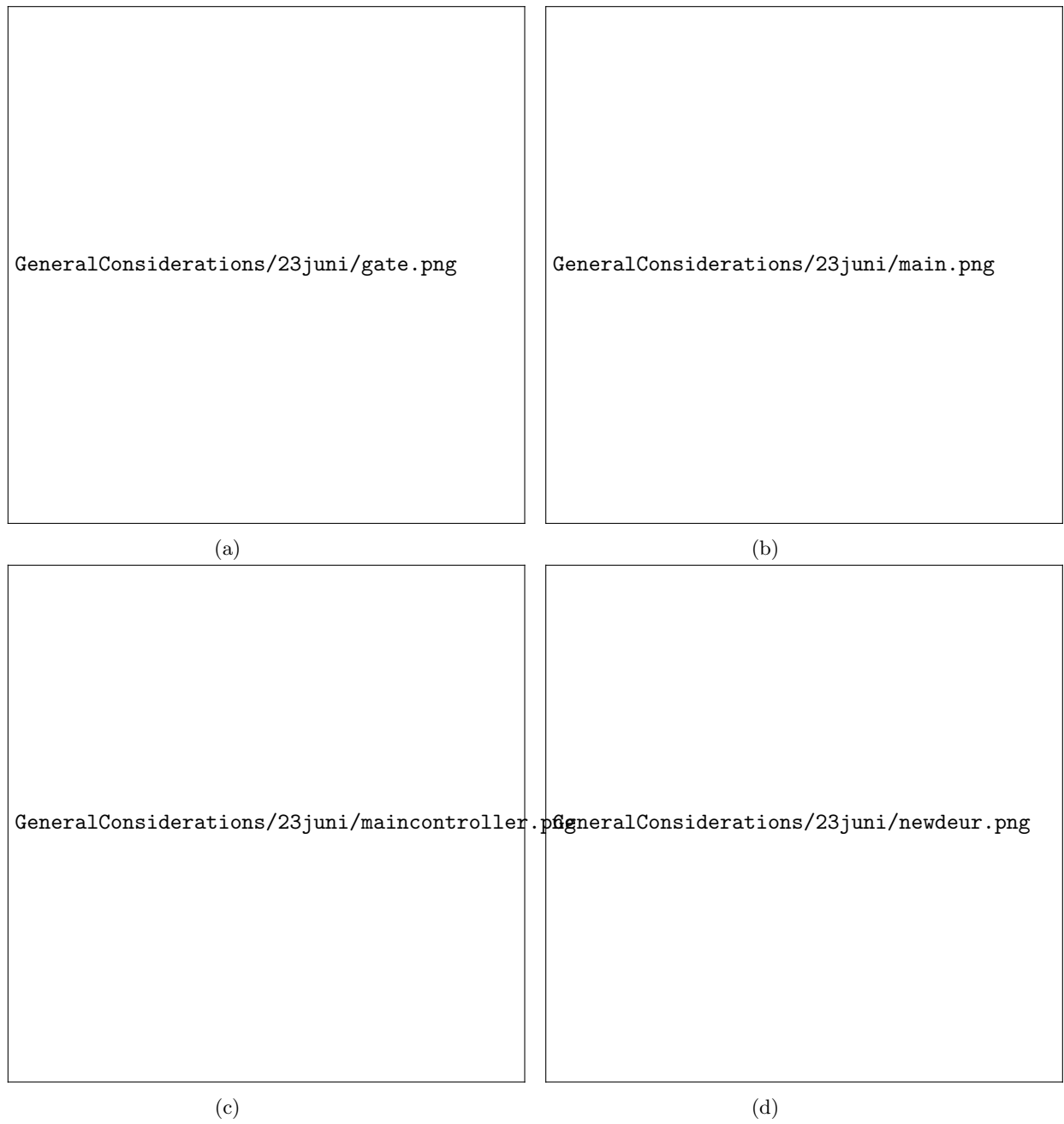
B.18 Ontwerpen: 23 juni



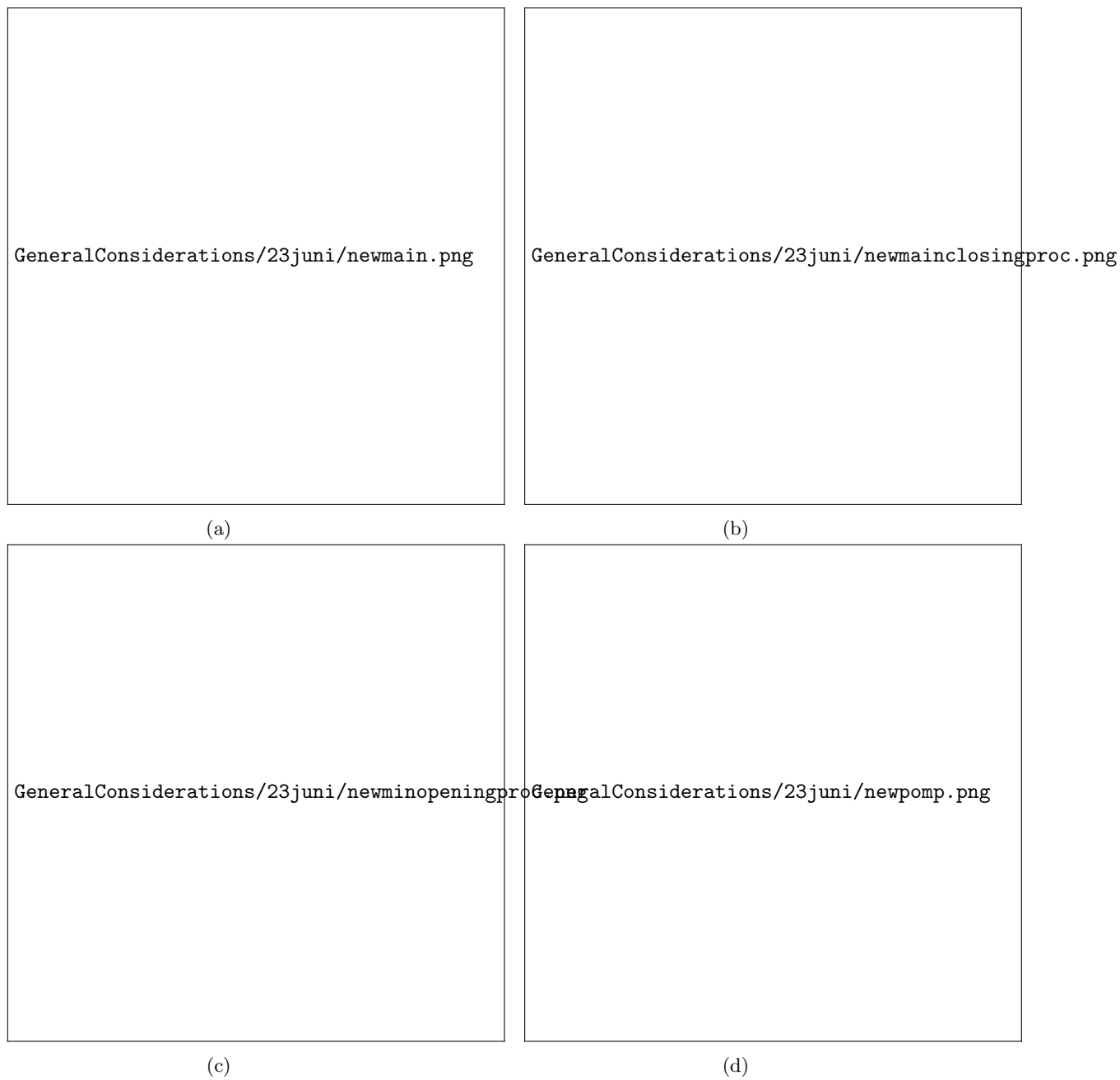
GeneralConsiderations/23juni/waterlevelsensoorHigh.png



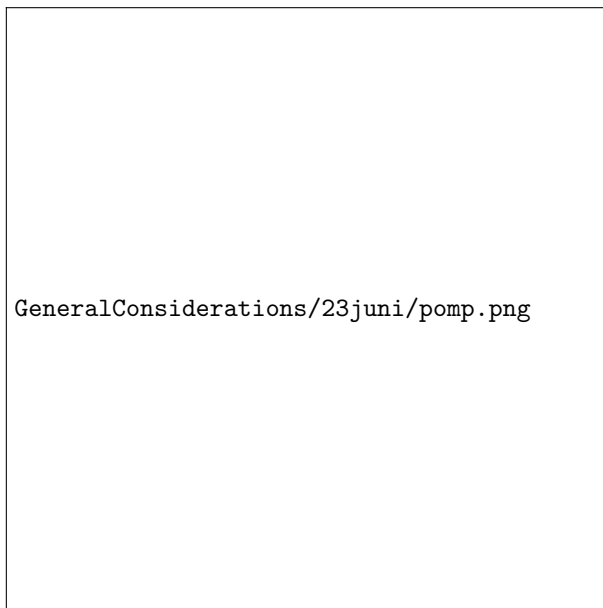
Figuur 59: Sluismodel 23 juni



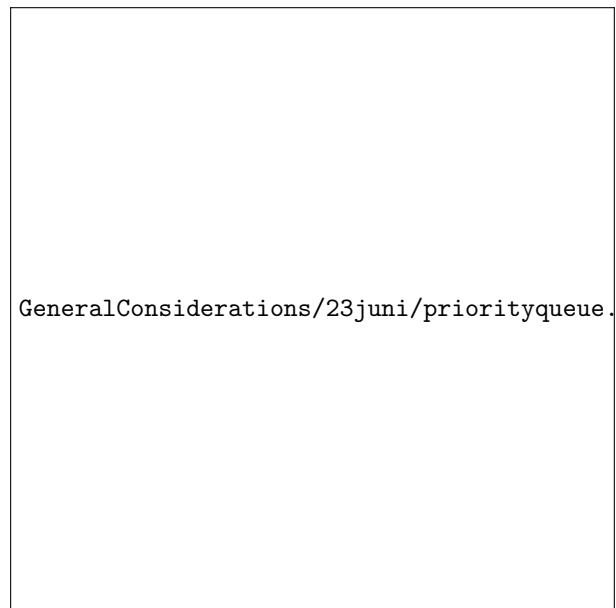
Figuur 60: Sluismodel 23 juni



Figuur 61: Sluismodel 23 juni



(a)



(b)



(c)

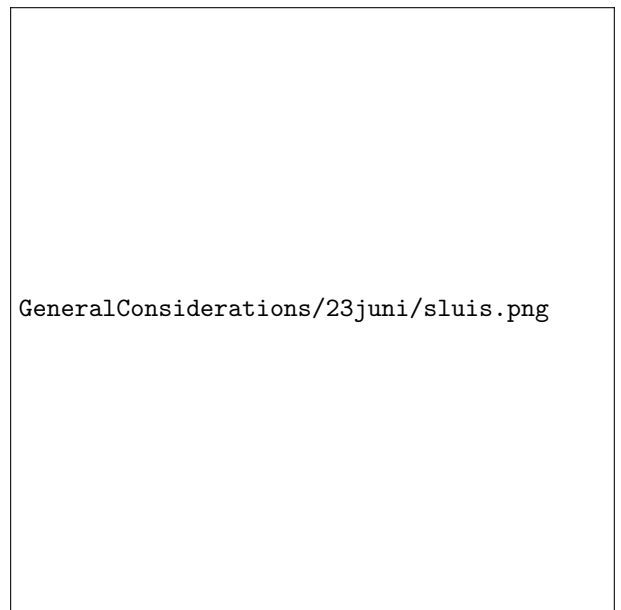


(d)

Figuur 62: Sluismodel 23 juni



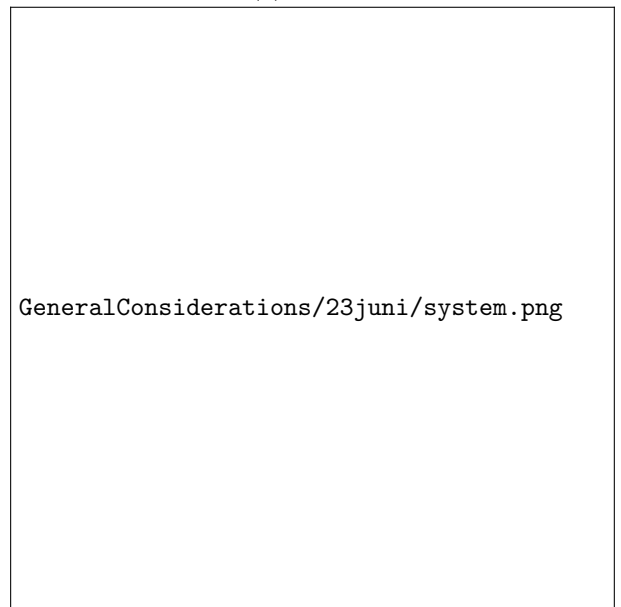
(a)



(b)



(c)



(d)

Figuur 63: Sluismodel 23 juni

B.19 Ontwerpen: 26 april



(a)



(b)

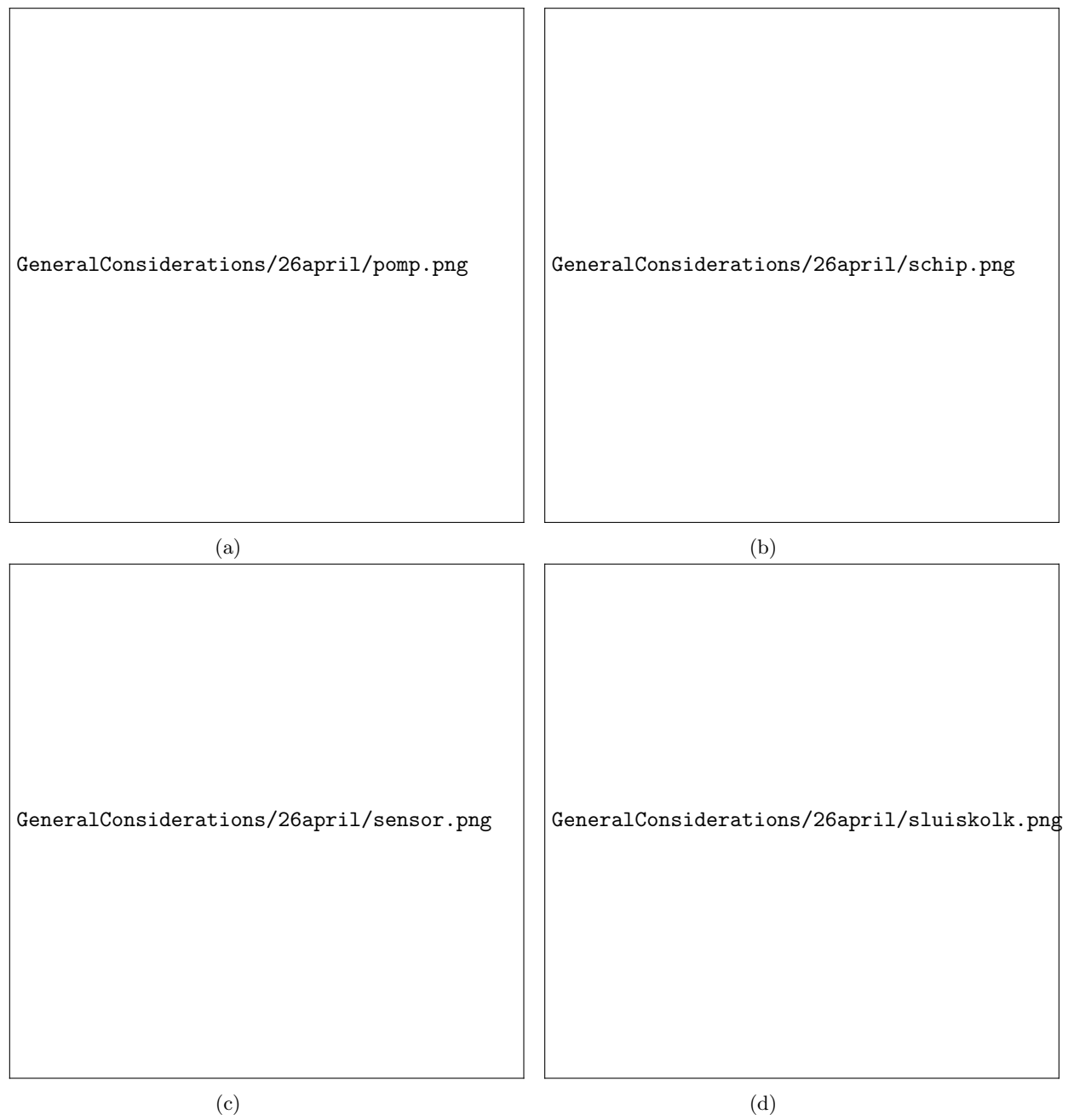


(c)



(d)

Figuur 64: Sluismodel 26 april



Figuur 65: Sluismodel 26 april



(a)



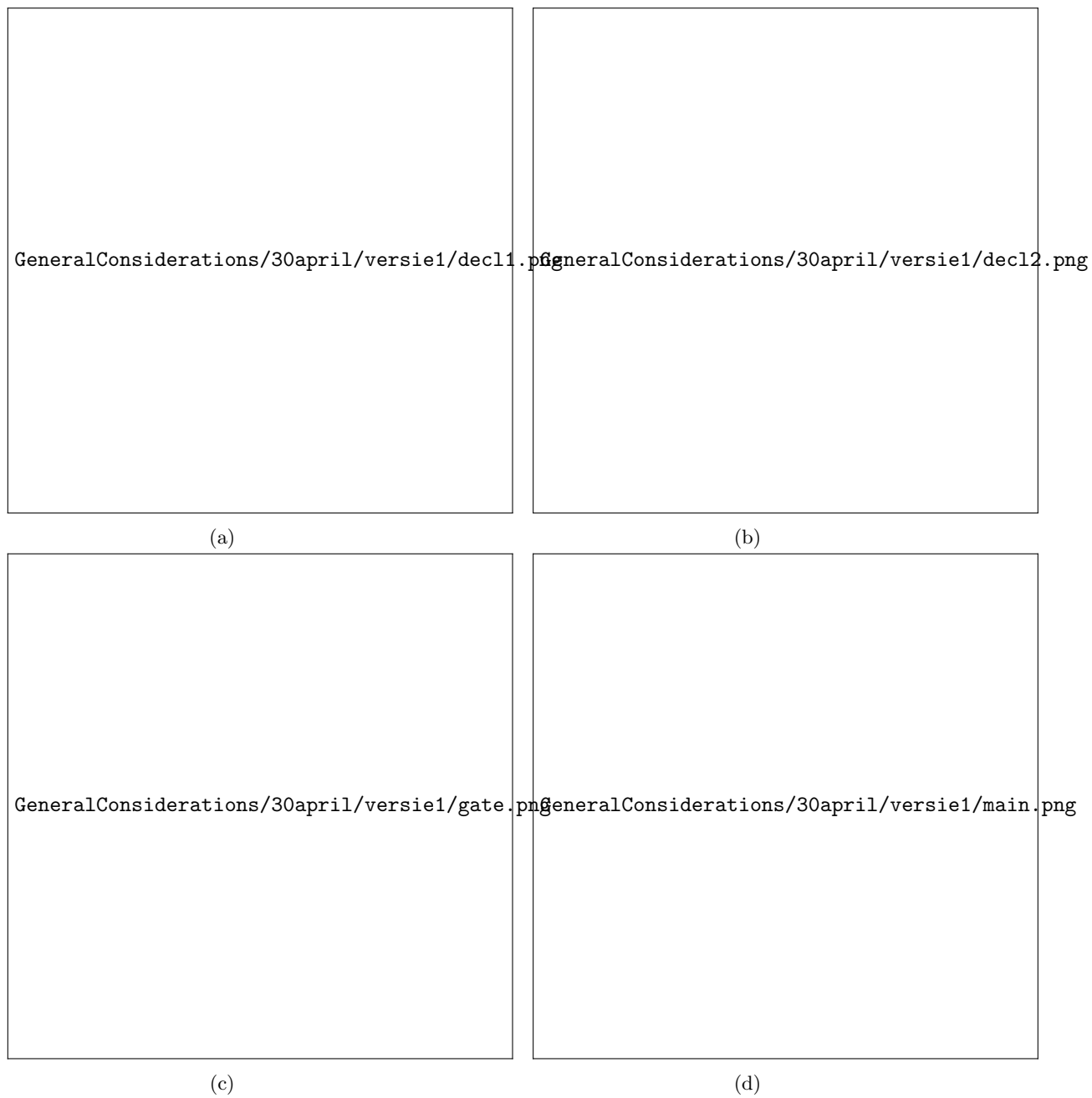
(b)

B.20 Ontwerpen: 30 april versie 1

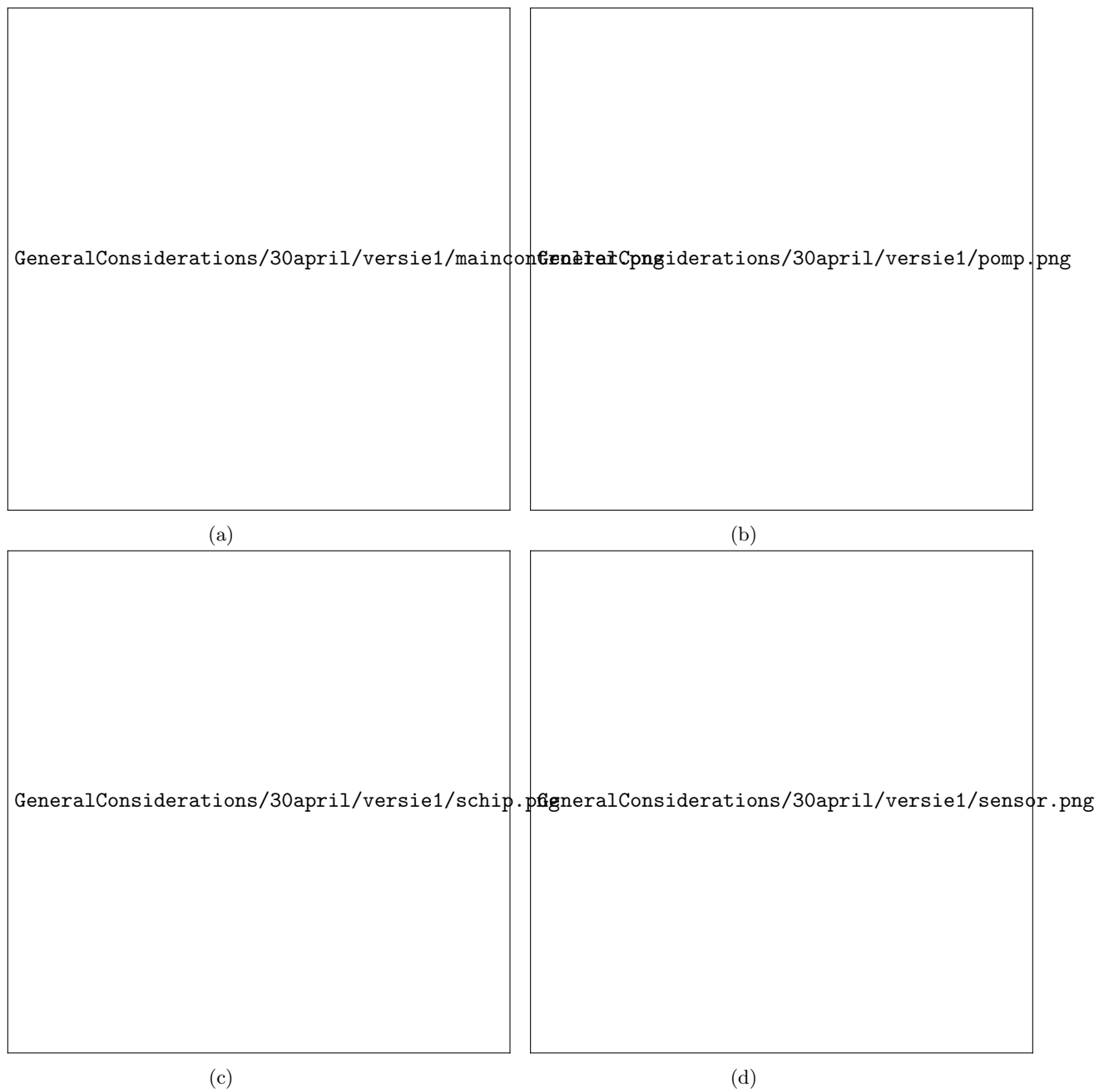




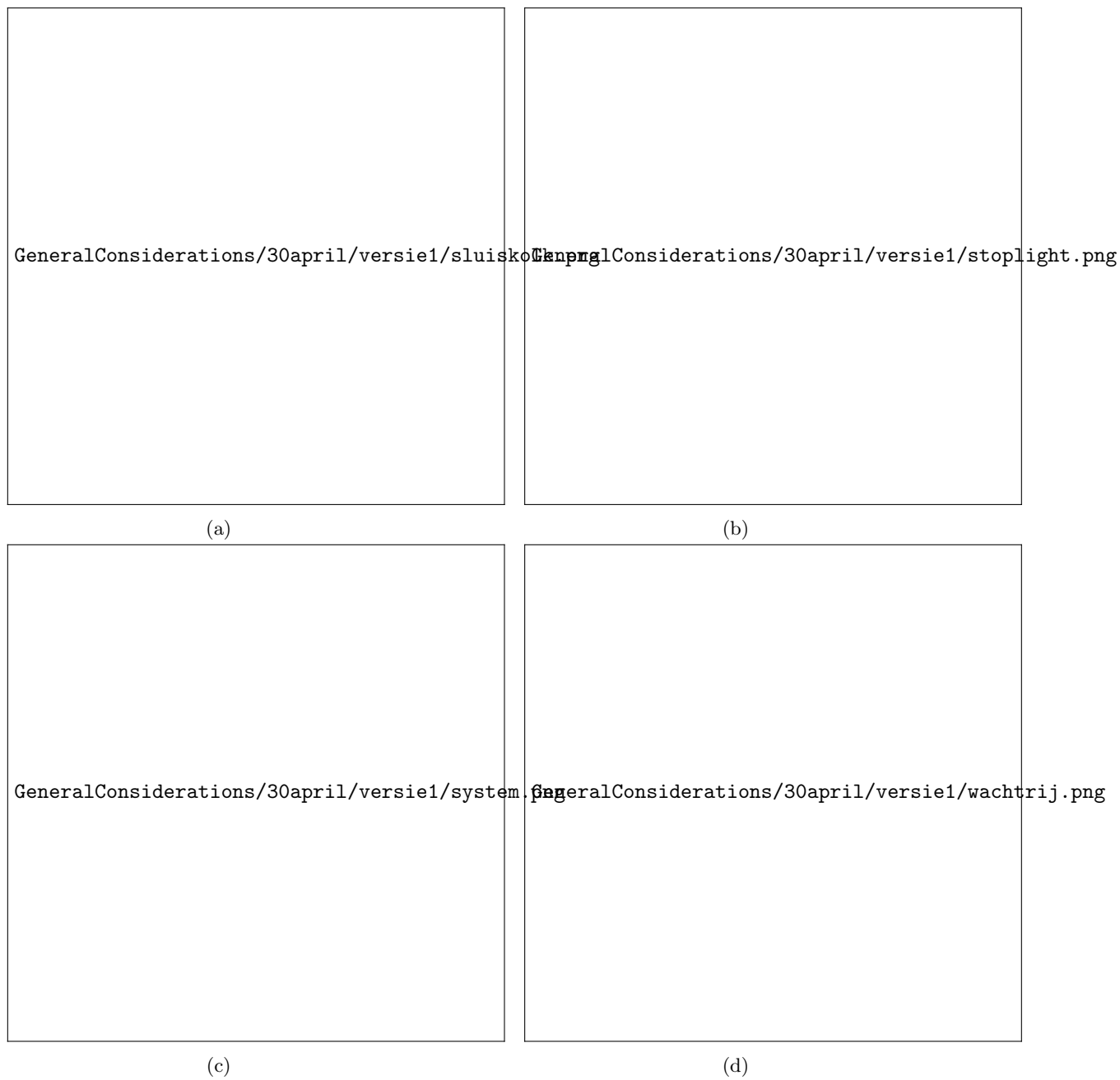




Figuur 67: Sluismodel 30 april



Figuur 68: Sluismodel 30 april



Figuur 69: Sluismodel 30 april