

2.1 Pizza or Pasta

Module ALICE:

Init:

$custPizzaBuffer = \emptyset$
 $custPastaBuffer = \emptyset$
 $pizzaBuffer = \emptyset$
 $pastaBuffer = \emptyset$

upon $\langle Bob, getJob \mid NULL \rangle$
 while $(pizzaBuffer == \emptyset) \{ \}$
 $dish \leftarrow pizzaBuffer$
 trigger $\langle Bob, prepareDish \mid dish \rangle$

upon $\langle Carole, getJob \mid NULL \rangle$
 while $(pastaBuffer == \emptyset) \{ \}$
 $dish \leftarrow pastaBuffer$
 trigger $\langle Carole, prepareDish \mid dish \rangle$

upon $\langle Customer, orderDish \mid customer, menu \rangle$ do
 if $(menu.sortOf(Pizza))$
 $custPizzaBuffer = custPizzaBuffer \cup customer$
 $pizzaBuffer = pizzaBuffer \cup menu$
 else
 $custPastaBuffer = custPastaBuffer \cup customer$
 $pastaBuffer = pastaBuffer \cup menu$

upon $\langle Bob, returnDish \mid menu \rangle$ do
 $customer \leftarrow custPizzaBuffer$
 $custPizzaBuffer = custPizzaBuffer \setminus customer$
 trigger $\langle Alice, serveDish \mid customer, menu \rangle$

upon $\langle Carole, returnDish \mid menu \rangle$ do
 $customer \leftarrow custPastaBuffer$
 $custPastaBuffer = custPastaBuffer \setminus customer$
 trigger $\langle Alice, serveDish \mid customer, menu \rangle$

Module BOB:

Init:

$bobPizzaBuffer = \emptyset$
 $triggerCounter = 0$

while (TRUE)
 if $|bobPizzaBuffer| + triggerCounter < 3$
 trigger $\langle Bob, getJob \mid NULL \rangle$
 $triggerCounter++$
 if $(|bobPizzaBuffer| \neq 0)$
 $dish \leftarrow bobPizzaBuffer$
 prepare the dish
 $bobPizzaBuffer = bobPizzaBuffer \setminus dish$
 trigger $\langle Bob, returnDish \mid dish \rangle$

upon $\langle Bob, prepareDish \mid dish \rangle$
 $bobPizzaBuffer = bobPizzaBuffer \cup dish$
 $triggerCounter-- = 1$

Module CAROLE:

Init:

$carolePastaBuffer = \emptyset$
 $triggerCounter = 0$

while (TRUE)
 if $|carolePastaBuffer| + triggerCounter < 7$
 trigger $\langle Carole, getJob \mid NULL \rangle$
 $triggerCounter++$
 if $(|carolePastaBuffer| \neq 0)$
 $dish \leftarrow carolePastaBuffer$
 prepare the dish
 $carolePastaBuffer = carolePastaBuffer \setminus dish$
 trigger $\langle Carole, returnDish \mid dish \rangle$

upon $\langle Carole, prepareDish \mid dish \rangle$
 $carolePastaBuffer = carolePastaBuffer \cup dish$
 $triggerCounter-- = 1$

2.2 Safety and liveness

- (a) *If some general attacks at time t , then the other general attacks at the same time.*
This is a ***liveness property***, because it says that if one of the two generals attacks the other general will attack with him, so eventually something good will happen.
- (b) *If m_2 arrives after time t , then General A attack after General B.*
This is a ***safety property***, because this statement ensures that General A will attack if and only if it receives the message delivered by m_2 and not just out of thin air.
- (c) *Eventually, General B will attack.*
This is a ***safety property***, because it ensures that if messenger m_1 has delivered the message (nothing bad has happened before), General B will attack.
- (d) *If messengers m_1 and m_2 are not intercepted, then eventually both generals attack.*
This is a ***safety property***, because it says that if nothing bad has happened in the past that the generals will attack at time t or directly after they received the message.
- (e) *If m_1 and m_2 are not intercepted, then eventually both generals attack at time t .*
This is a ***mixture***, because we have the safety property that both messenger will arrive in time - so before time t - and the liveness property that eventually both attack at the same time t .

2.3 Unreliable clocks

- (a) *Find two examples, where timing issues lead to safety violations.*
- (b) *Find two examples, where timing issues lead to liveness violations.*