Proces rozproszony

Plan wykładu

Celem wykładu jest zapoznanie słuchacza z podstawowymi pojęciami związanymi z przetwarzaniem rozproszonym. Wykład ten jest kontynuacją wykładu poprzedniego, w którym zdefiniowano podstawowe pojęcia związane z przetwarzaniem rozproszonym. Niniejszy wykład obejmie w pierwszej części omówienie m. in. definicji procesu rozproszonego, wykonania, globalnego stanu spójnego czy też historii realizacji. Następnie zdefiniowana i omówiona zostanie relacja poprzedzania zdarzeń, a także zaprezentowane i omówione zostaną diagramy przestrzenno-czasowe. W kolejnej części wykładu przedstawione zostaną pojęcia przetwarzania niedeterministycznego i grafu stanów osiągalnych. Przedostatnim tematem poruszonym w trakcie tego wykładu będzie mechanizm monitora procesu. Na zakończenie krótko omówiona zostanie ujednolicona konwencja zapisu algorytmów, które będą prezentowane na kolejnych wykładach.

Proces rozproszony

Proces rozproszony Π, będący współbieżnym wykonaniem zbioru $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ procesów sekwencyjnych P_i, opisuje uporządkowana czwórka $\Pi = \langle \boldsymbol{\Sigma}, \boldsymbol{\Sigma}^0, \boldsymbol{\Lambda}, \boldsymbol{\Phi} \rangle$, gdzie:

$\boldsymbol{\Sigma}$ **–** jest zbiorem stanów globalnych procesu rozproszonego, $\boldsymbol{\Sigma} \subseteq \mathcal{S}_1 \times \mathcal{S}_2 \times ... \times \mathcal{S}_n$,

$\boldsymbol{\Sigma}^0$ – jest zbiorem stanów początkowych, $\boldsymbol{\Sigma}^0 \subseteq \mathcal{S}_1^0 \times \mathcal{S}_2^0 \times ... \times \mathcal{S}_n^0$,

$\boldsymbol{\Lambda}$ – jest zbiorem zdarzeń, $\boldsymbol{\Lambda} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup ... \cup \mathcal{E}_n$;

$\boldsymbol{\Phi}$ **–** jest funkcją tranzycji, taką że $\boldsymbol{\Phi} \subseteq \boldsymbol{\Sigma} \times \boldsymbol{\Lambda} \times \boldsymbol{\Sigma}$.

Zbiór stanów globalnych (1)

Zbiór stanów globalnych $\boldsymbol{\Sigma}$ jest podzbiorem iloczynu kartezjańskiego $\mathcal{S}_1 \times \mathcal{S}_2 \times ... \times \mathcal{S}_n$ zbiorów stanów lokalnych procesów składowych przetwarzania rozproszonego. Stan globalny procesu rozproszonego Π w chwili τ czasu globalnego (czasu postrzeganego przez teoretycznego zewnętrznego obserwatora), oznaczony przez $\Sigma(\tau)$, jest więc uporządkowanym zbiorem stanów lokalnych wszystkich procesów składowych w chwili τ:

$$\Sigma(\tau) = \langle S_1(\tau), S_2(\tau), ..., S_n(\tau) \rangle$$

Zbiór stanów globalnych (2)

Zbiór globalnych stanów początkowych $\boldsymbol{\Sigma}^0$ jest podzbiorem iloczynu kartezjańskiego $\mathcal{S}_1^0 \times \mathcal{S}_2^0 \times ... \times \mathcal{S}_n^0$ zbiorów stanów początkowych procesów składowych.

Zbiór zdarzeń globalnych $\boldsymbol{\Lambda}$ jest sumą mnogościową $\mathcal{E}_1 \cup \mathcal{E}_2 \cup ... \cup \mathcal{E}_n$ zbiorów zdarzeń procesów składowych.

Globalna funkcja tranzycji

Globalna funkcja tranzycji Φ jest złożeniem funkcji tranzycji procesów składowych, a więc jest zbiorem trójek $\langle \Sigma, E, \Sigma' \rangle$, dla których istnieje takie k, $1 \leqslant k \leqslant n$, że $\boldsymbol{\Sigma} = \langle S_1, S_2, ..., S_k, ..., S_n \rangle$, $\boldsymbol{\Sigma}' = \langle S_1, S_2, ..., S'_k, ..., S_n \rangle$, oraz $\langle S, E, S' \rangle \in \mathcal{F}_k$.

Wykonanie częściowe procesu

Zajście zdarzenia w którymkolwiek z procesów implikuje zmianę jego stanu, a tym samym zmianę stanu globalnego. Ponieważ zdarzenia są z założenia atomowe, można przyjąć, że w każdej chwili zachodzi co najwyżej jedno zdarzenie. Stąd też, **Częściowe wykonanie** procesu rozproszonego $\Pi = \langle \boldsymbol{\Sigma}, \boldsymbol{\Sigma}^0, \boldsymbol{\Lambda}, \boldsymbol{\Phi} \rangle$ utożsamia się z ciągiem $\Sigma^0, E^1, \Sigma^1, E^2, ... , \Sigma^s, E^{s+1}, \Sigma^{s+1}$, składającym się naprzemiennie ze stanów i zdarzeń, takim że dla każdego u, $0 \leqslant u \leqslant s$, $\langle \Sigma^u, E^{u+1}, \Sigma^{u+1} \rangle \in \boldsymbol{\Phi}$.

Wykonanie procesu

Przez **wykonanie** (**realizację**) Υ procesu Π rozumieć będziemy częściowe wykonanie rozpoczynające się stanem początkowym $\Sigma^0 \in \boldsymbol{\Sigma}^0$.

Stan osiągalny

Powiemy, że stan Σ' procesu jest osiągalny ze stanu Σ, co oznaczymy przez $\Sigma \trianglerighteq \Sigma'$, jeżeli istnieje częściowe wykonanie $\Sigma^0, E^1, \Sigma^1, E^2, ... , \Sigma^s, E^{s+1}, \Sigma^{s+1}$ procesu Π, takie że $\Sigma = \Sigma^0$, a $\Sigma' = \Sigma^{s+1}$.

Globalny stan osiągalny

Oznaczmy przez $\boldsymbol{\Upsilon}$ zbiór wszystkich możliwych wykonań (realizacji) procesu Π. Jeżeli istnieje wykonanie procesu Π, takie że Σ jest stanem końcowym, to stan ten nazwiemy **globalnym stanem osiągalnym** (**spójnym**) procesu rozproszonego Π (ang. *reachable*, *consistent*).

Historia wykonania

Każdemu wykonaniu $\Upsilon \in \boldsymbol{\Upsilon}$ procesu Π, odpowiada pewien ciąg stanów $\Sigma^0, \Sigma^1, \Sigma^2, ... , \Sigma^s, \Sigma^{s+1}$, nazywany **śladem wykonania (realizacji) procesu** Π, oraz ciąg zdarzeń $E^0, E^1, E^2, ..., E^s, E^{s+1}$, nazywany **historią wykonania (realizacji) procesu**. Historię $E^0, E^1, E^2, ..., E^s$ oznaczamy przez Ξ^s, a zbiór historii – przez $\boldsymbol{\Xi}$.

Podobnie jak dla procesu sekwencyjnego, kolejne zdarzenia E^u, E^{u+1} realizacji procesu Π określają interwał czasu globalnego, a stan procesu Π w tym interwale definiuje dotychczasowa historia Ξ^u realizacji procesu. W efekcie stan Σ^u można utożsamiać z historią Ξ^u, i powiedzieć, że zdarzenia historii Ξ^u należą do stanu Σ^u.

Proces rozproszony jako graf

Proces rozproszony jest często przedstawiany, podobnie jak środowisko rozproszone, jako graf niezorientowany lub zorientowany

$$\mathcal{G} = \langle \mathcal{V}, \mathcal{A} \rangle$$

w którym wierzchołki grafu $V_i \in \mathcal{V}$ reprezentują procesy składowe $P_i \in \mathcal{P}$ przetwarzania rozproszonego, a krawędzie $(V_i, V_j) \in \mathcal{A}$, $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V}$, grafu niezorientowanego lub łuki $\langle V_i, V_j \rangle \in \mathcal{A}$ grafu zorientowanego, reprezentują kanały $C_{i,j}$.

Topologia przetwarzania rozproszonego

Tak zdefiniowany graf jest **grafem procesu rozproszonego** lub **topologią przetwarzania** (**topologią procesu rozproszonego**).

Relacja poprzedzania zdarzeń

Zbiór zdarzeń $\mathcal{E}_i$ procesu P_i jest **w pełni uporządkowany**, według kolejności ich występowania w czasie lokalnym τ. Ze względu jednak na nieprzewidywalne czasy transmisji i brak wiedzy o czasie globalnym τ, uporządkowanie w praktyce zbioru Λ zdarzeń wszystkich procesów $P_i \in \mathcal{P}$, stanowi poważną trudność.

Oznaczmy przez $\mapsto$ **relację poprzedzania** (ang. *happen before, causal precedence, happened before*) zdefiniowaną na zbiorze $\boldsymbol{\Lambda}$ w następujący sposób:

$E_i^k \mapsto E_j^l \Leftrightarrow$ {
1) $i = j \wedge k < l$, lub
2) $i \neq j$ oraz E_i^k jest zdarzeniem $e\_send(P_i, P_j, M)$ wysłania wiadomości M, a zdarzenie E_j^l jest zdarzeniem $e\_receive(P_i, P_j, M)$ odbioru tej samej wiadomości, lub
3) istnieje sekwencja zdarzeń $E^0, E^1, E^2, ... , E^s$, taka że $E^0 = E_i^k$, $E^s = E_j^l$ i dla każdej pary $\langle E^u, E^{u+1} \rangle$ gdzie $0 \leqslant u \leqslant s$ -1, zachodzi 1) albo 2).

Relacja poprzedzania jest **antysymetryczna** i **przechodnia**, a więc jest **relacją częściowego porządku**.

Relacja poprzedzania lokalnego

Przez $\mapsto_l$ oznaczamy **relację poprzedzania lokalnego** zdarzeń procesu P_i, taką że: $E_i^k \mapsto_l E_j^l$ wtedy i tylko wtedy, gdy $i=j$ oraz $k<l$ (lub gdy $i=j$ oraz $E_i^k \mapsto E_j^l$).

Zdarzenia przyczynowo-zależne

Zdarzenia E_i^k i E_j^l nazywamy **przyczynowo-zależnymi**, jeżeli: $E_i^k \mapsto E_j^l$ albo $E_j^l \mapsto E_i^k$. W przeciwnym razie zdarzenie te nazwiemy **przyczynowo-niezależnymi** lub **współbieżnymi** (ang. *concurrent, causally independent*), co będziemy oznaczać przez $E_i^k \parallel E_j^l$.

Diagramy przestrzenno-czasowe

Realizację przetwarzania rozproszonego można przedstawić graficznie w postaci **diagramu przestrzenno-czasowego** (ang. *space-time diagram*), w którym osie reprezentują upływ czasu globalnego τ, a punkty na osiach – zdarzenia.

Przykładowy diagram

Slajd przedstawia przykładowy diagram przestrzenno-czasowy. Na diagramie tym są pokazane różne rodzaje zależności między zdarzeniami zachodzącymi dla różnych procesów. Pierwszy typ zależności to zdarzenia współbieżne. Cztery przykładowe zależności przedstawione na slajdzie to $E_1^1 \parallel E_2^1$, $E_1^1 \parallel E_2^2$, $E_1^1 \parallel E_3^1$, oraz $E_1^1 \parallel E_3^4$. Kolejne przykłady pokazują zdarzenia przyczynowo zależne, gdzie relacja ta związana jest bezpośrednio z przesłaniem wiadomości aplikacyjnych. Trzy takie przykładowe pary to $E_3^1 \mapsto E_1^2$, $E_3^3 \mapsto E_2^3$, oraz $E_1^4 \mapsto E_2^4$. Kolejne przykłady pokazują zdarzenia, które nie są bezpośrednio przyczynowo zależne, natomiast można znaleźć ciąg zdarzeń, którego efektem jest relacja przyczynowej zależności. Trzy przykładowe pary zdarzeń to $E_3^1 \mapsto E_1^4$, $E_3^1 \mapsto E_2^3$, oraz $E_3^2 \mapsto E_2^4$.

Relacja poprzedzania stanów lokalnych

Przez analogię do relacji na zbiorze zdarzeń, można zdefiniować częściowy porządek na zbiorze stanów wszystkich procesów $P_i \in \mathcal{P}$ w sposób następujący:

$$S_i^k \mapsto S_j^l \Leftrightarrow \begin{cases} E_i^{k+1} \mapsto E_j^l, \text{ lub} \\ E_i^{k+1} = E_j^l \end{cases}$$

Relacja powyższa oznacza, że stan jednego procesu poprzedza przyczynowo stan innego, wtedy i tylko wtedy, gdy zdarzenie rozpoczynające drugi stan zależy przyczynowo bądź jest tożsame ze zdarzeniem kończącym pierwszy stan.

Stany współbieżne

Stany lokalne, dla których nie zachodzi ani relacja ani $S_i^k \mapsto S_j^l$ ani też relacja $S_j^l \mapsto S_i^k$, nazywamy **współbieżnymi**. Zauważmy, że jeżeli w danym stanie Σ są dopuszczalne (aktywne) dwa różne zdarzenia E^k i E^l, to kolejnym stanem będzie stan Σ', jeżeli $\langle \Sigma, E^k, \Sigma' \rangle \in \boldsymbol{\Phi}$ i zaszło zdarzenie E^k. Kolejnym stanem będzie natomiast Σ'', jeżeli $\langle \Sigma, E^l, \Sigma'' \rangle \in \boldsymbol{\Phi}$ i zaszło zdarzenie E^l. Ponieważ ograniczenia na kolejność wystąpienia zdarzeń określa relacja poprzedzania, to zbiór $\boldsymbol{\Lambda}$ zdarzeń wraz z relacją poprzedzania stanowi zbiór częściowo uporządkowany $\langle \boldsymbol{\Lambda}, \mapsto \rangle$ opisujący możliwe **realizacje przetwarzania**.

Rozważmy dla przykładu relację poprzedzania zdefiniowaną przez diagram przestrzenno-czasowy przedstawiony powyżej. Przyjmując stan początkowy $\Sigma^0 = \langle H_1^0, H_2^0, H_3^0 \rangle$, stwierdzamy,

że zdarzenia E_2^1 i E_3^1 są współbieżne, więc po stanie $\Sigma^0 = \langle H_1^0, H_2^0, H_3^0 \rangle$ możliwy jest stan $\Sigma^1 = \langle H_1^0, H_2^1, H_3^0 \rangle$ albo $\Sigma^2 = \langle H_1^0, H_2^1, H_3^1 \rangle$. Postępując tak dalej, możemy określić następne stany.

Graf stanów osiągalnych

Zbiór częściowo uporządkowany $\langle \mathbf{\Lambda}, \mapsto \rangle$ może być przedstawiony w postaci grafu zorientowanego, w którym wierzchołki odpowiadają stanom Σ, a łuki $\langle \Sigma^k, \Sigma^l \rangle$ oznaczają istnienie zdarzenia dopuszczalnego E takiego, że $\langle \Sigma^k, E, \Sigma^l \rangle \in \mathbf{\Phi}$. Graf taki, będziemy nazywać **grafem stanów osiągalnych przetwarzania rozproszonego** lub **siatką obliczeń rozproszonych**.

Przykład grafu stanów osiągalnych

Dla ilustracji tego zagadnienia, rozważmy graf stanów osiągalnych przetwarzania rozproszonego odpowiadający diagramowi przestrzenno-czasowemu, a więc określony w pełni przez relację poprzedzania między zdarzeniami wynikającą wyłącznie z diagramu, przyjmując jednak, że zdarzenia współbieżne mogą zachodzić w dowolnej kolejności względem siebie.

Na slajdzie przedstawiono przykładowy diagram przestrzenno-czasowy oraz odpowiadający mu graf stanów osiągalnych. Dla uproszczenia oznaczono tu stan $\langle S_1^i, S_2^j, S_3^k \rangle$ przez $\Sigma^{i,j,k}$.

Ponieważ kolejność występowania zdarzeń współbieżnych jest dowolna, relacja poprzedzania określa (modeluje) wiele możliwych realizacji, z których każda odpowiada pewnej ścieżce w grafie stanów osiągalnych przetwarzania. Z drugiej strony, relacja ta określa jednoznacznie zależności przyczynowe między zdarzeniami lokalnymi, w tym oczywiście – między zdarzeniami odbioru przez określony proces wiadomości wysłanych w wyniku zdarzeń współbieżnych. Zauważmy jednak, że zdarzenia odbioru wiadomości M_1 i M_2 przedstawione na przykładowym diagramie, mogą w ogólności zajść w odwrotnej kolejności, jeżeli tylko zdarzenie odbioru w procesie docelowym jest P_1 wynikiem operacji niedeterministycznego odbioru $receive(\mathcal{P}_1^S, P_1, sInM)$, gdzie $\mathcal{P}_1^S = \{P_2, P_3\}$ i nadejście wiadomości M_2 wyprzedzi nadejście wiadomości M_1.

Uwzględnienie takiej alternatywy w modelu przetwarzania rozproszonego, jakim jest zbiór częściowo uporządkowany $\langle \mathbf{\Lambda}, \mapsto \rangle$, jest oczywiście możliwe przez stosowną zmianę zbioru $\mathbf{\Lambda}$ i definicji relacji poprzedzania. Dlatego też w literaturze zbiór uporządkowany $\langle \mathbf{\Lambda}, \mapsto \rangle$ przyjmuje się często za ogólny model przetwarzania, reprezentujący pewien zbiór możliwych jego realizacji.

Niedeterminizm przetwarzania

W kontekście grafu stanów osiągalnych przetwarzania rozproszonego, każda realizacja przetwarzania rozproszonego jest pewną ścieżką w tym grafie. Istnienie wielu różnych ścieżek ilustruje **niedeterminizm** przetwarzania rozproszonego, oznaczający że dla danego stanu może istnieć wiele stanów następnych.

Niedeterministyczne zdarzenie lokalne

Powiemy, że lokalne zdarzenie procesu jest **niedeterministyczne**, gdy jego zajście może być zastąpione przez zajście innego zdarzenia i wybór ten nie jest przewidywalny. Jeżeli przykładowo sekwencyjne wykonanie procesu może być w każdej chwili zmienione w wyniku zajścia przerwania zewnętrznego, to wszystkie zdarzenia tego procesu są niezdeterminowane. Podobnie, wykonanie alternatywnych struktur sterujących w językach CSP czy ADA jest

równoważne niedeterminizmowi. Innym źródłem niedeterminizmu jest asynchroniczny dostęp do zasobów w systemach wielowątkowych i przełączanie procesora.

W przetwarzaniu rozproszonym najbardziej charakterystyczny jest niedeterminizm zdarzeń odbioru. Wynika on z faktu, że relatywne prędkości procesów są nieznane, a czasy transmisji są skończone ale nieprzewidywalne. Stąd wykonanie operacji odbioru $receive(\mathcal{P}_1^S, P_1, sInM)$ może prowadzić do różnych zdarzeń (np. odbioru wiadomości od różnych nadawców).

Przetwarzanie zdeterminowane i niedeterministyczne

Przetwarzanie nazywamy **zdeterminowanym** jeżeli wszystkie zdarzenia są zdeterminowane. W przeciwnym wypadku, przetwarzanie nazywamy **niedeterministycznym**.

Przetwarzanie quasi-deterministyczne

W ramach niedeterministycznego przetwarzania rozproszonego wyróżnia się podklasę przetwarzania **quasi deterministycznego** (ang. *quasi-deterministic*, *piece-wise deterministic*, *event-driven*), w której niedeterminizm jest wyłącznie konsekwencją niedeterminizmu operacji odbioru.

Diagramy równoważne

Należy zauważyć, że w ogólności istnieje wiele różnych diagramów przestrzenno-czasowych, którym odpowiada taki sam zbiór częściowo uporządkowany $\langle \mathbf{\Lambda}, \mapsto \rangle$. Diagramy takie nazywa się **diagramami równoważnymi**. Diagram przestrzenno-czasowy może być odwzorowany w diagram równoważny przez ,,rozciąganie" lub ,,ściskanie" linii reprezentujących lokalne osie czasu, jeżeli w wyniku tych operacji relacja poprzedzania dowolnej pary zdarzeń zależnych nie zostanie zmieniona.

Przykład diagramów równoważnych

Dwa przykładowe równoważne diagramy przestrzenno-czasowe przedstawione są na slajdzie.

Monitor

Ocena wartości predykatów globalnych wymaga w pierwszej kolejności obserwacji (monitorowania) stanów lokalnych procesów składowych. W tym celu przyjmiemy, że z każdym procesem P_i skojarzony jest **proces monitora** Q_i. Relacje między procesem P_i i jego monitorem Q_i przedstawiono na slajdzie.

Cechy monitora

Monitor Q_i może odczytywać (obserwować) zmienne lokalne procesu P_i, a więc określać stan lokalny procesu. Monitor nie ma natomiast możliwości zmiany stanu procesu przez przypisanie jego zmiennym lokalnym nowych wartości. Ponadto, monitor może obserwować i kontrolować zdarzenia komunikacyjne. Kontrola ta polega na uzupełnianiu wiadomości wysyłanych przez P_i o dodatkową informację sterującą adresowaną do monitora procesu docelowego, oraz na przechwytywaniu wiadomości skierowanych do procesu i interpretacji zawartej w nich informacji sterującej. W efekcie, wiadomości nadchodzące mogą być zatrzymane na pewien okres przez monitor i przekazane procesowi docelowemu dopiero, gdy spełnione zostaną określone warunki. Przyjmiemy, że przekazywanie wiadomości M przez monitor Q_i procesowi docelowemu P_i odpowiada wykonaniu operacji $deliver(P_j, P_i, M)$, gdzie P_j jest nadawcą wiadomości M. Operacja ta powoduje uaktywnienie zdarzenia odbioru $e\_receive(P_j, P_i, M)$. Zakładamy przy tym, że stymulowane operacjami $deliver(P_j, P_i, M)$ zdarzenia $e\_receive(P_j, P_i, M)$ zachodzą w kolejności wykonywania operacji $deliver(P_j, P_i, M)$.

W ogólności, monitory nie muszą być wyróżnione jako specjalne procesy, gdyż ich zadanie może pełnić oprogramowanie wbudowane w proces aplikacyjny, zgodnie z regułami superpozycji oprogramowania. Takie podejście nazywa się czasami instrumentacją. Tak więc wyróżnienie monitorów jest w istocie jedynie sprawą interpretacji i implementacji.

Konwencja zapisu algorytmów

Przyjęto, że celem prezentacji algorytmów jest możliwie precyzyjne przedstawienie ich ogólnej idei a nie pełnej implementacji. Dlatego do specyfikacji algorytmów posłużono się notacją, wywodzącą się z powszechnie znanych języków wysokiego poziomu oraz z symboliki matematycznej. Ponadto, wszędzie tam, gdzie było to możliwe, pominięto szczegóły implementacyjne, wprowadzając w ich miejsce opisy słowne.

Dla ujednolicenia prezentacji, a jednocześnie zaznaczenia ról jakie pełnią różne komunikaty w poszczególnych algorytmach, wyróżnione zostały następujące ich typy: komunikaty aplikacyjne, komunikaty kontrolne, sygnały i pakiety. **Komunikaty aplikacyjne** przekazywane są bezpośrednio lub za pośrednictwem monitorów między procesami aplikacyjnymi, działającymi w różnych węzłach systemu. Przenoszą one dane istotne dla aplikacji rozproszonej. **Komunikaty kontrolne** przekazywane są między monitorami w celu koordynacji ich działań lub detekcji określonych stanów. Nie są one związane bezpośrednio z przetwarzaniem aplikacyjnym. **Sygnały** są z kolei komunikatami kontrolnymi, które nie przenoszą żadnych danych poza podstawowymi atrybutami każdego komunikatu. W końcu, **pakiety** są komunikatami przekazywanymi pomiędzy monitorami różnych węzłów systemu. Przenoszą one komunikaty aplikacyjne oraz dodatkowe dane, które są istotne z punktu widzenia koordynacji działań monitorów.

Wszystkie komunikaty, niezależnie od ich typu, mają pewne wspólne atrybuty: identyfikator typu komunikatu, identyfikator komunikatu, identyfikator nadawcy oraz identyfikator odbiorcy. W celu opisu struktury przekazywanych komunikatów przyjmujemy formalizm, zgodny z koncepcją dziedziczenia cech znaną z języków obiektowych. Każdy komunikat ma pewną strukturę wywiedzioną z innej struktury.

typ FRAME

Podstawową strukturą, z której pośrednio lub bezpośrednio wywodzą się struktury wszystkich innych komunikatów jest `FRAME`. Obejmuje ona wymienione wyżej wspólne atrybuty komunikatów. Zakłada się ponadto, że struktura komunikatu jest identyfikowana przez jej nazwę, w związku z czym nie musi być ona jawnie wyróżniany jako atrybut. Z punktu widzenia języka, `FRAME` jest typem danych zdefiniowanym w sposób następujący:

```
type FRAME is record of
      tag: ...
      mId: ...
      sId: ...
      rId: ...
end record
```

typ MESSAGE

Struktura komunikatów aplikacyjnych, nazwana `MESSAGE`, jest wywiedziona ze struktury `FRAME`, a jej precyzyjna definicja zawarta jest w programie aplikacji rozproszonej i nie jest istotna z punktu widzenia prezentowanych dalej algorytmów. Przyjmiemy zatem tylko, że szkielet definicji struktury `MESSAGE` ma następującą postać:

```
type MESSAGE extends FRAME is record of
      ...
end record
```


typ CONTROL

Komunikat kontrolny ma strukturę, zwaną `CONTROL`, która jest wywiedziona ze struktury `FRAME`. Precyzyjna definicja struktury `CONTROL` jest zależna od algorytmu, a jej szkielet jest następujący:

```
type CONTROL extends FRAME is record of
      ...
end record
```

typ SIGNAL

Sygnał jest komunikatem, który nie przenosi żadnych dodatkowych danych. Jego struktura, nazwana `SIGNAL`, jest zatem ,,pustym" rozwinięciem struktury `FRAME`. Ewentualne dalsze rozwinięcia struktury `SIGNAL` nie wnoszą przy tym żadnego nowego atrybutu, a jedynie zmieniają wartość niejawnego identyfikatora typu – *tag*. Stąd:

```
type SIGNAL extends FRAME is record of
      ...
end record
```

typ PACKET

Jak już wspomniano, przed przekazaniem komunikatu aplikacyjnego do środowiska komunikacyjnego może on być uzupełniany o dane istotne dla algorytmu realizowanego przez monitory, w wyniku czego powstaje pakiet. Szkielet definicji struktury pakietu, nazwanej `PACKET`, ma postać przedstawioną poniżej. Właściwa struktura przesyłanych pakietów, zdefiniowana jest w ramach konkretnego algorytmu. Tak więc:

```
type PACKET extends FRAME is record of
      ...
      data: MESSAGE;
end record
```