```
        async def stream(self, query_in: str,
                         context_id: str) -> AsyncGenerator[dict[str, Any], None]:
            # 解析フェーズ
            yield {
                "is_task_complete": False,
                "require_user_input": False,
                "content": "入力を解析しています…",
            }

            # 簡易的なコンテキストの維持
            if context_id not in self.context_id2query:
                self.context_id2query[context_id] = ""
            self.context_id2query[context_id] += query_in + "¥n"
            query = self.context_id2query[context_id]

            c_from, c_to, date_raw, parse_err = self._extract_params(query)
            if parse_err:
                yield {
                    "is_task_complete": False,
                    "require_user_input": True,
                    "content": f"入力解析に失敗しました: {parse_err}¥n"
                               "例: `USD EUR`"
                               "/ `usd jpy 2024-06-01` "
                               "/ `米ドルをユーロに 先週の金曜`",
                }
                return

            missing = []
            if not c_from:
                missing.append("通貨（変換元, 3文字コード）")
            if not c_to:
                missing.append("通貨（変換先, 3文字コード）")

            if missing:
                yield {
                    "is_task_complete": False,
                    "require_user_input": True,
                    "content": (
                        "為替レートを取得するために次の情報が必要です："
                        f"{' と '.join(missing)}。¥n"
                        "例: `USD EUR` や `usd jpy 先週の金曜`"
                    ),
                }
                return

            if c_from == c_to:
                yield {
                    "is_task_complete": True,
                    "require_user_input": False,
                    "content": f"{c_from} と {c_to} は同一通貨です。為替レートは常に 1.0 です。",
                }
                return

            # 日付正規化（モデル解釈）
            yield {
                "is_task_complete": False,
                "require_user_input": False,
                "content": f"日付を解釈しています…（指定: {date_raw or 'latest'}）",
            }
            date_norm, date_warn = self._normalize_date(date_raw or "latest")

            # レート取得
            yield {
                "is_task_complete": False,
                "require_user_input": False,
                "content": f"為替レートを取得中…（{c_from} → {c_to}, 日付: {date_norm}）",
            }

            data = get_exchange_rate(c_from, c_to, date_norm)

            # 整形
            yield {
                "is_task_complete": False,
                "require_user_input": False,
                "content": "結果を整形しています…",
            }

            if "error" in data:
                msg = f"取得に失敗しました：{data['error']}¥n"
                if date_warn:
                    msg += f"補足: {date_warn}¥n"
                msg += "通貨コード（例: USD EUR）と、必要なら自然言語で日付を指定して再実行してください。"
                yield {
                    "is_task_complete": False,
                    "require_user_input": True,
                    "content": msg,
                }
                return
```

```
089
090             rate = data.get("rates", {}).get(c_to)
091             if rate is None:
092                 yield {
093                     "is_task_complete": False,
094                     "require_user_input": True,
095                     "content": "指定の通貨コードが不正か、レートが取得できませんでした。¥n"
096                                "例: `USD EUR 2024-06-01` や `USD EUR last Friday`。",
097                 }
098                 return
099
100             date_used = data.get("date", date_norm)
101             inv = 1.0 / rate if rate else None
102
103             lines = [
104                 f"【為替レート】{c_from} → {c_to}",
105                 f"- 日付: {date_used}",
106                 f"- 1 {c_from} = {rate:.6f} {c_to}",
107             ]
108             if inv:
109                 lines.append(f"- 1 {c_to} = {inv:.6f} {c_from}")
110             if date_warn:
111                 lines.append(f"¥n注: {date_warn}")
112
113             yield {
114                 "is_task_complete": True,
115                 "require_user_input": False,
116                 "content": "¥n".join(lines),
117             }
```